The University Of Sheffield.

DESIGN AND REAL-TIME CONTROL OF A NEW STRUCTURE TWO-WHEELED ROBOT

by

Omar Khalil Sayidmarie

A thesis submitted to the University of Sheffield for the degree of
Doctor of Philosophy

Department of Automatic Control and Systems Engineering
The University of Sheffield
Mappin Street
Sheffield, S1 3JD
UK

February 2016

Dedication

To the memory of my beloved grandmother who sent 'her son' to England like his father for further education, and she was waiting for his successful return. But when I finished, she passed away this life.

ABSTRACT

The work presented in this thesis deals with the design and real-time implementation of a control system for a new configuration of two-wheeled robot. In real life applications, two wheeled robots are considered to carry payloads of different sizes at different positions and different motion speeds along the vertical axis. Such parameters will have an impact on the robot stability and the control mechanism, hence their detailed study is essential for robust performance of the system. This work investigates the impact of the dynamic change of the payload position on the system damping characteristics whilst the robot is in its balancing state.

A mathematical model is developed to describe and study the dynamics of the two-wheeled robot system. Accurate tilt angle measurement is achieved through applying a sensor fusion realized by complementary filter. A PID control strategy is considered to balance and position the robot assuming a fixed location of the payload. Then the controller is extended to provide the robot with the ability of self-standing from its rest position without human interaction. The developed controller and the sensor fusion filter are implemented on a microcontroller development board. The results show that the implementation of the controller fulfils the requirement to balance the robot.

The two-wheeled robot test rig was modified to accommodate the payload actuation unit and improve the overall performance. Consequentially, the controller of the robot is also modified with the sensor fusion algorithm enhanced by implementing Kalman filter. The controller comprise a PID feedback with a feedforward approach. Furthermore, gain scheduling approach is utilized to ensure smooth and fast braking of the two-wheeled robot

The control approach is extended to an intelligent controller, where a PD-like fuzzy controller is design. A binary coding technique is developed for real-time implementation of the fuzzy controller. Such coding and implementation eliminates the need to store a big lookup table for the controller rules. The controller is tested in laboratory experiments and its robustness is demonstrated with application of various disturbance forces on the system.

In order to evaluate the performance of the new configuration two-wheeled robot, various experiments are conducted under different conditions. The results demonstrate that the proposed two-wheeled robot configuration and the proposed control approaches form a solid foundation and a framework for assisted mobility of disabled and elderly people.

ACKNOWLEDGEMENTS

All praise and glory are due to Almighty Allah, the sole Lord of the universe, whose mercy and blessings have been bestowed constantly upon me. Peace and blessings be upon His messenger Mohammad.

First of all, I would like to thank my supervisors, Doctor Osman Tokhi, for his valuable advice and guidance throughout my PhD program. I am sincerely grateful an indebted to him for giving me the opportunity to prove myself. I am honoured to have had the privilege to work with him. He is an inspirational scientist and a role model. I would also like to thank all members of the research group for useful discussions. I would like to thank my sponsor the Iraqi Ministry of Higher Education and Scientific Research (MOHESR) for the financial support of my study. This work would have not been possible without the support and encouragement of my Family. So I would like to thank each and every one of my family members for always believing in me. I especially would like to thank my parents, aunts, and my wife for their support and looking after me. Last but not least, I would like to thank my friends for putting up with me throughout my studies.

.

Table of Contents

List of Figures

List of Tables

Nomenclature

Terminology	*Description*	*Units*
$\dot{\theta_m}$	The angular velocity of the motor	*rad/sec*
$\ddot{\theta_m}$	The angular acceleration of the motor	*rad/sec²*
$\dot{\delta}$	Angular speed of the wheel	*rad/sec*
a	Complementary filter coefficient	
Acc_x	Accelerometer x-axis output in volts	*V*
Acc_y	Accelerometer y-axis output in volts	*V*
Acc_z	Accelerometer z-axis output in volts	*V*
b	the friction constant of the motor	*Nm.sec/rad*
D	The travelled distance of the robot	*m*
dt	Sampling time	*sec*
e	The back electromotive force emf	*V*
$e(t)$	Error signal	
F	Linear driving force	*N*
F_d	External disturbance force	*N*
g	Gravitational acceleration (9.81)	*m/sec²*
$G(s)$	The low pass filter transfer function	
i_a	The current of the motor	*A*
J	Moment of inertia	*kg.m²*
J_I	Moment of inertia of the chassis	*kg.m²*
J_i	The instantaneous moment of inertia of the robot	*kg.m²*
J_w	Moment of inertia of the wheel	*kg.m²*
K_d	Derivative gain	
K_i	Integral gain	
K_K	Kalman gain matrix	
K_m	The motor (torque) constant	*Nm.sec/rad*
K_p	Proportional gain	
K_v	The velocity constant of the motor	*V.sec/rad*
l	Distance to the centre of mass	*m*
L	Lagrangian function	
L_a	The inductance of the motor's coil	*H*
l_t	The global location of COM of the robot	*m*
M	Mass of the payload	*Kg*
m	Mass of the chassis	*Kg*
m_I	The total mass of the chassis and the payload	*kg*
M_w	Mass of the wheel	*Kg*
N	The gear ratio of the motor	

P	Priori error covariance matrix	
Q	The distance of the payload from the original COM	m
Qi	The generalized force associated with a generalized coordinate	
qi	The generalized coordinate	
Q_k	The covariance of the process noise	
R	Radius of the wheel	m
R_a	The resistance of the motor's coil	Ohm
R_k	The covariance of the measurement noise	
S	The travel speed of the robot	m/sec
S	The innovation covariance matrix	
T	Total kinetic energy of the system	$kg.m^2/sec^2$
T_R	The rotation kinetic energy of the system	$kg.m^2/sec^2$
T_r	The torque of the motor's shaft	$N.m$
T_t	The translation kinetic energy of the system	$kg.m^2/sec^2$
V	Total potential energy of the system	$kg.m^2/sec^2$
v	The transitional, linear, velocity	m/sec
v_a	The applied voltage of the motor	V
V_k	Zero mean white measurement noise	
v_{offset}	The accelerometer output offset voltage	V
w	The angular velocity	rad/sec
W_k	Zero mean white process noise	
$X(k)$	Complementary filter input	
$X_{Acc}(k)$	The accelerometer data	
$X_{Gy}(k)$	The gyroscope data	
$Y(k)$	Complementary filter output	
Z_k	The observation or measurement	
α	The roll angle	rad
δ	Angular displacement of the wheel	m
Δt	Sampling time	sec
θ	Tilt angle of the chassis	rad
τ	The time constant of the filter	sec
$u(t)$	Control signal	

Abbreviations

Abbreviation	*Description*
ADC	Analogue to digital Convertor
ANFIS	Adaptive neural network fuzzy
CAN	Controller area network
CCW	Counter clockwise
COG	Centre of gravity
COM	Centre of mass
CPLD	Complex programmable logic device
CW	Clockwise
DAC	Digital to analogue convertor
DC	Direct current
DOF	Degrees of freedom
DSP	Digital signal processing
emf	Back electromotive force
EMU	Embedded microprocessor unit
FLC	Fuzzy logic controller
FPGA	Field programmable gate array
GPS	Global positioning system
GUI	Graphic user interface
I^2C	Inter-IC
IMU	Inertial measurement unit
IR	Infra-red
LQR	Linear quadratic regulator
MAE	Mean absolute error
MCU	Microcontroller unit
MEMS	Microelectromechanical sensors
MF	Membership functions
MIPS	Mega instruction per second

MSPS	Mega sample per second
NB	Negative big
NN	Neural-network
NS	Negative small
OS	Operating system
PB	Positive big
PC	Personal computer
PD	Proportional derivative
PI	Proportional integral
PID	Proportional integral derivative
PS	Positive small
PUMA	Personal urban mobility and accessibility
PWM	Pulse width modulation
RBFNNs	Radial basis function neural networks
RCT	Reference compensation technique
RF	Radio frequency
RPM	Revolution per minute
RSME	Root mean squared error
SBC	Single board computers
SMC	Sliding mode control
SPI	Serial peripheral interface
TS	Takagi-sugeno
TV	Television
TWR	Two – wheeled robot
UART	Universal synchronous/asynchronous receiver/transmitter

CHAPTER 1

Introduction and literature review

1.1 Introduction

Two-wheeled robots, which operate on the inverted pendulum principle, are in nature unstable, non-linear, and under actuated. Basically, the robot consists of two driving wheels attached on each side of the robot chassis and driven by two dc motors. Just as the inverted pendulum, two wheeled robots are balanced at upright position by controlling the rotation of the wheels or in other words the rotation of the dc motors. Since the introduction of the two wheeled robot JOE (Grasser et al. 2002), this kind of robots has induced great interest as they provide opportunities for application of control design, signal processing, distributed control systems and consideration of real-time constraints during implementation. Moreover, they offer advantages over other ground robotic vehicles such as zero-turn radius, simple structural design, and small footprint for navigation in tight spaces. However, owing to the inherent instability, the system needs to be dynamically controlled, and sensors need to be installed on the system for manoeuvring. Because of these intriguing requirements, two wheeled robots are widely used in research institutions to study control techniques with relevant sensors. In recent years, researchers have applied the idea of a mobile inverted pendulum model to various problems like designing walking gaits for humanoid robots, robotic wheelchairs and personal transport systems.

Commercial products, based on two wheeled robot, have also been developed, such as the Segway human transporter, PUMA (Personal Urban Mobility and Accessibility) (Kamen 2001), and the motorized wheelchair iBot (Takahashi et al. 2001). The nature of such two wheeled vehicle poses several interesting control questions. For instance, while a person occupies the vehicle, his/her mass changes the centre of gravity of the vehicle which in turn has an impact on the control technique used. One of the recent challenges in such applications is the development of control schemes to help a disabled or an elderly person on a two wheeled wheelchair, as shown in the iBot wheelchair Figure 1.1, to move to further levels in

shopping centres or to have eye to eye contact with others. Industrial applications of such vehicles will arise to a great extent in the coming years; for instance, material handling to different heights in narrow pathways.

Figure 1.1: ***iBot wheelchair offers high shelf reach (http://www.hizook.com/blog/2009/02/11/ibot-discontinued-unfortunate-disabled-perhaps-budding-robotics-opportunity)***

1.2 Literature review

1.2.1 Control strategies

Two wheeled robots are non-linear systems; however the most common controllers used in the literature are linear due to their simple implementation process. New approaches in non-linear control systems which have been encouraged by developments in neural networks and fuzzy logic are potential candidates in this area. Many researchers have suggested that intelligent control approaches will improve the robustness and control stability for the two wheeled robots, and some have suggested hybrid control approaches.

Grasser et al. (2002), have built JOE, a prototype of a two-wheeled vehicle based on the inverted pendulum concept. The main objective of this vehicle is to balance its driver on two coaxial wheels. The authors neglected the motor dynamics, due to their small time constant, in developing the mathematical model of the system and linearized the model around the operating point. The control system was based on two decoupled state-space controllers: one balancing the robot at the upright position, while the second one acting on the steering around the vertical axis (yaw).

System performance (i.e., reaction to disturbance forces, tracking of driver input, etc.) was assisted with pole placement, and controllers with different pole placements were tested to maximize JOE's performance. Three types of disturbances that indicate the movement made by the driver were also included in the system.

Pathak et al. (2005) have analysed the dynamic model of a two wheeled inverted pendulum robot from a controllability and feedback points of view. They derived the dynamic model of the under actuated system; the system has a lower number of actuators than degrees of freedom; with respect to the wheel motor torques as its inputs by taking the non-holonomic no-slip constraints into consideration. The partial feedback linearization of the system was obtained based on the results of the accessibility condition and the maximum relative degree of the system. The resulting equations were used to design two novel controllers of which one of them was a two-level velocity controller used for tracking vehicle orientation and heading speed set-points, while controlling the vehicle pitch. The other controller which was also a two-level controller used to stabilize the vehicle's position to the desired set-point, while ensuring the pitch to be bounded between specified limits.

Jingtao et al. (2007) have designed a two-wheeled inverted pendulum mobile robot to be low power, lightweight and compact. The robot has a high manoeuvrability advantage to move on inclined surfaces. The authors provided a dynamic model of the robot and presented the equations of motion via linear transformation of the derived nonlinear equations. The robot had multi-mode control system which includes the human transport control and goods transport control mode. In the human transport mode the reference for a proportional integral derivative (PID) controllers is generated from the sensors conditioning module, whereas in the second mode the reference is sent by the operator over wireless link. Experiments have been carried out with the robot to test its ability to climb an inclined plane and to avoid obstacles. Authors have claimed that the robot successfully moved on an inclined surface with an inclination of 15^{o} and avoided small obstacles.

Coelho et al. (2008) have presented a mobile platform based on the two wheeled inverted pendulum concept for indoor use. This robot was designed to have

the ability to test autonomous applications such as obstacle avoidance and people following. Two controllers were designed and implemented using MATLAB and Simulink. The balance and speed controller was linear quadratic regulator (LQR) type based on a linear model of the system, while simple PID controller was used for steering control.

Similarly, a single wleeled inverted pendulum vehicle has been presented in (Huang 2010). The vehicle was mathimaticaly modeled based on Lagrange's equation and the resulted nonlinear eqution were linearzed around the operating piont. The author uesd LQR controller to overcome the difficulties in ajusting the gains in pole-placement and state-feedback controls and improve the stibilty of the vehicle.

Hyung et al. (2010) have developed a controller for a two wheeled inverted pendulum robot system called TransBOT with the capability of carrying a human operator. Two driving modes were developed: mode I is a regular driving mode of wheeled mobile robots. Mode II is the balancing mode that mimics the inverted pendulum concept by lifting up the front casters. The authors designed three separate linear controllers which used for the pendulum angle, the orientation angle and the cart position. One proportional derivative (PD) controller for the angle control and two PID controllers for the orientation and position control were used. Driving tests have been conducted in indoor and outdoor environments. Test results confirm that the feasibility and practicality of the developed TransBOT.

Some researchers utilized energy-shaping controllers to control inverted pendulum systems. An energy-shaping controller has been presented by Angeli (2001). The author developed an almost global stabilisation of the inverted pendulum on cart system. The author made use of energy shaping with a smooth switching feedback control between positive and negative feedback signals to stabilize the inverted pendulum system.

Intelligent and hybrid controllers have been investigated and developed for various types of inverted pendulum-based systems. Some hybrid-type controllers combined intelligent control algorithms such as fuzzy logic and neural networks with

a classic PID controller, whereas some other combined energy-shaping controllers with sliding mode controllers.

Marzi (2005) has presented the stages of development of a fuzzy logic based controller for an inverted pendulum over cart by developing a two-input, Mamdani type system. The author compared and analysed the influence of increasing the number of inputs on the complexity of the controller and designing time. The author has also studied the effects of number of membership functions in a four input Takagi-Sugeno (TS) type on the stability of the system. The author claim that the four inputs TS fuzzy controller with five membership functions per input is very complex and performance is slow, but if tuned correctly, a system of this size would be very precise compared to a similar system with two membership functions per input.

A self-tuning PID control strategy, based on a deduced model, has been proposed in (Ren et al. 2008) for implementing a motion control system that stabilizes a two wheeled vehicle and follows the desired motion commands. In this work the authors have developed a neural-network (NN)-like self-tuning PID control scheme in order to improve the stability and the ability to quickly recover from any disturbance and parameter variation in the two wheeled robot system. The controller consists of two self-tuning PIDs for position control and balancing of the robot.

Jung and Kim (2008) have presented an online learning and control using NN for a wheel-driven mobile inverted pendulum robot. The NN has been used as reference compensation technique (RCT) to control the system along with primary PID controllers by compensating for uncertainties in dynamics in an online adaptive fashion. The developed NN included six inputs, nine hidden units, and six output units to compensate for each component of the two PIDs, to track the desired position of the robot and maintain its upright balance.

Tsai et al. (2010) have presented an adaptive control approach using radial basis function neural networks (RBFNNs) for a self-balancing two-wheeled scooter. The system was mathematically modelled and linearized in a state-space framework with two unknown frictions. The system was decomposed into two subsystems, and two adaptive controllers using RBFNN were synthesized to achieve self-balancing

and desired yaw motion. The objectives of the two RBFNNs were to approximately learn viscous and static frictions between the wheels and the motion surface online, uncertainties of riders' weights, and un-modelled errors caused by linearization. The authors showed, through simulations and experimental results, that the proposed controllers were useful and effective in providing appropriate control actions to steer the vehicle at slow speeds.

Junfeng and Wanying (2011) have derived a mathematical model of the GBOT1001, two wheeled inverted pendulum robot produced by Googol Technology Limited, based on Newton dynamics mechanics theory. They assumed that the robot is rigid, the wheels are completely analogous, cornering forces are negligible, no friction to be included, and the motors dynamics are neglected due to their small time constant compared to the system's time constant. The authors designed a hybrid controller for the system which is constructed from classic PD controller for balancing and fuzzy logic based PD like controller with seven membership functions for position control of the robot. The performance of the system was tested using Simulink / MATLAB, and it was shown that the proposed hybrid fuzzy PID controller can successfully control a two wheeled robot. Also Junfeng et al. (2011) have used the same mathematical model to develop a robust controller for the robot based on sliding mode control (SMC) algorithm to achieve robust stabilization and disturbance rejection of the GBOT1001 robot.

In another work of Junfeng and Shengwei (2011) they have simulated a TS adaptive neural network fuzzy controller (ANFIS) for the two wheeled robot GBOT1001. The training data of the controller were taken from the input and output data of a model, derived previously, with a pole placement controller. The trained ANFIS model was applied to control the robot.

In the same area of intelligent control of the two wheeled robot, Wang and Fang (2014) presented a fuzzy immune PD algorithm to control self-balancing robots. The results of their research showed that the new technique offers advantages over the classical controllers in terms of stability and smaller overshoot, and shorter adjustment time.

Prasad et al. (2011) have presented two hybrid controllers for the inverted pendulum over cart problem to control the position of the cart and balance the pendulum in the presence of disturbances, and both structures were simulated in MATLAB. The first was a combination of a classic PID controller for cart position and a LQR for balancing. While the second structure from two PID controllers for cart position and balancing were combined negatively with LQR controller. The authors observed that the response of the system with hybrid PID + LQR controller was better than with PID control. They observed that the system responses with the two hybrid controllers, 2 PID + LQR and PID + LQR, were similar.

Another way of designing a hybrid controller of the inverted pendulum robots is reported in (Fukushima et al. 2015) where a hybridization between sliding mode non-linear controller and LQR linear controller is presented. The main aim of the controller is to control the transformation process from a four-wheeled mode for high-speed mobility to an inverted pendulum mode. The performance and the effectiveness of the developed controller were evaluated in both simulation and practical experiment.

Evolutionary algorithms for optimizing controllers of inverted pendulum systems have also been investigated.

Ahmad et al. (2009) have applied genetic optimization algorithm on a fuzzy logic based controller (FLC) for lifting and balancing a two-wheeled wheelchair. The genetic algorithm was used to find the optimum values for the input-output scaling factors of the FLC. Results of the simulation have shown the effectiveness of the approach.

Genetic optimization algorithm has been applied in (Goher and Tokhi 2009; 2010) to find the best PID gain values for the steering, position, balancing, and payload lifting controller of a two wheeled robot with an additional degree of freedom along its vertical axis.

1.2.2 Real time implementation

In terms of real time implementation, digital circuits such as microcontroller (MCU), field programmable gate array (FPGA) chips, digital signal processing (DSP) devices, and single board computers, have been used to implement the inverted pendulum control systems. Generally, the experimental real time implementation methods of the two wheeled robots in the literature can be classified into two parts. The first comprises self-contained systems, and the second part outlines the real time PC based systems. On the other hand, some researchers have used the real-time toolbox of MATLAB to implement the controller of an inverted pendulum system.

Accurate current state sensing is an important issue in developing a good controller for the two wheeled inverted pendulum. Such measurements could be done using an Inertial Measurement Unit (IMU), but it is relatively large and expensive. Some researchers have tried to utilize a potentiometer with a rod, either touching the ground or as balancing rod to achieve tilt angle (Grant 2009), while some have reported the use of distance sensors such as infra-red (IR) and ultra sound to estimate the tilt angle of the two wheeled inverted pendulum robot (Hassenplug 2002). Inclination sensors such as inclinometers, accelerometers and gyroscopes have been used to measure tilt angle in deferent ways (Jung and Kim 2008). Some researchers suggest methods and techniques for combining two or more different sensor types to achieve accurate tilt angle measurement (Nawawi et al. 2008; Ren et al. 2008).

JOE in (Grasser et al. 2002) was one of the first self-contained two wheeled inverted pendulum robots implemented in real time. It was controlled by a linear state space controller utilising sensory information from a single gyroscope and two incremental encoders mounted on the motors. The controller was implemented on DSP development board, which was composed of a Sharcfloating-point DSP, a XILINX field-programmable gate array (FPGA), four 10-bit D/A converters, and fourteen 12-bit A/D converters. The time drift phenomenon in the tilt angle measurement was described by a ramp-shaped disturbance angle applied to the system, and this disturbance was compensated by the speed controller.

Kim and Jung (2006) have presented the hardware implementation of a neural network controller for a non-linear two wheeled robot with a microcontroller unit

(MCU) and an FPGA chip. They used the MCU to implement the neural network control algorithm, and designed the FPGA chip for the embedded PIDs and reading the sensors. A communication block was designed and built in the FPGA chip which received PID gains and desired trajectory from the MCU, and then transferred encoded data whenever it was needed. Later in another work, the authors used a DSP board with the FPGA chip to realize the real time implementation of a hybrid neural network controller for a non-linear inverted pendulum system (Jung and Kim 2007).

Leavitt et al. (2006) have outlined a method for high bandwidth tilt angle measuring technique by utilizing low cost sensors such as a pendulum-type inclinometer, accelerometer and gyroscope. Although, each one of these sensors has few properties that can cope with the two wheeled inverted pendulum robots. The authors constructed an optimal linear state estimator, based on Kalman filter, which estimates the gyroscope bias, and infers the accurate tilt angle from the output of all three sensors.

Jingtao et al. (2007) have implemented an embedded PID control system for a two wheeled robot. The micro-processor Em104- -i613, which runs RT-linux real time operating system, was chosen for signal processing and implementing the PID controllers for stabilizing and coordinating the robot. An I/O expansion module, A/D and D/A module were connected to the processor through the PC/104 bus to read the speed data of motors and various sensors information. The robot was equipped with a gyroscope combined with an accelerometer for the tilt angle estimation and two encoders of 2000 counts per revolution for measuring the rotation of the wheels. A wireless communication, which is used for transferring the operator's commands and monitoring data between the robot and PC, was realized by programming the RS232 bus on both sides and using a wireless RF module.

Nawawi et al. (2008) have derived a mathematical model of a two wheeled robot prototype using Newtonian mechanics, then linearized the result around the operating point. They designed a controller based on pole placement technique and analysed the results. Issues like damping ratio and settling time identified on the recorded responses and used for fine-tuning of the system. The controller was implemented on the prototype, in real time. The robot equipped with Gyroscope for

measuring angle and angular velocity of pendulum body, encoders for measuring the position of the wheels and an embedded DSP board. The embedded control system was designed based on real-time workshop of MATLAB and they used C language interfacing in the Real Time Workshop of MATLAB to send the command to the embedded controller. They claim that the pole placement controller is capable to balance the robot in terms of steady state error and settling time.

Ren et al. (2008) have implemented a self-tuning PID control strategy for a two wheeled robot. They used a DSP board to build two NN-like self-tuning PID to control the position of the robot and balance it in the upright. In order to measure the current state of the robot, yaw angle, tilt angle, and the angular position of the wheels, the authors used two gyroscopes, one accelerometer, and two encoders. The desired travelling velocity and steering angle can be sent to the robot through a wireless RF joystick.

Jean and Wang (2009) have built a prototype of a two-wheel vehicle with the considerations of cost-efficiency. The balancing of the vehicle was based on the fuzzy logic control with two inputs, single output and 25 fuzzy rules. The controller was realized using the SUNPLUSTM^{TM} embedded board (EMU) and a complex programmable logic device (CPLD). The EMU board is a low-cost 16-bit MCU with operating frequency of 49.152MHz, which is equipped with 32 programmable multi-functional I/Os, an eight-channel 10-bit ADC, two 10-bit DACs, and UART. The CPLD board was used to implement the pre-processing circuit of the tilt sensors and the PWM generator circuit which drives the motor.

Tsai et al. (2010) have used the TMS320LF2407 DSP board, from Texas Instruments, to implement two adaptive controllers using RBFNNs for two-wheeled self-balancing scooter. Gyroscope and tilt sensors were employed for measuring the rate and the angle of inclination of the footplate caused by the rider. Potentiometer was adapted to measure the yaw angle from the handlebar, and the yaw signal was then taken by the DSP controller for steering control.

Miasa et al. (2010) have reported the implementation of a two inputs single output fuzzy logic based controller with 10 rules to balance a two wheeled robot. The ADXL330 accelerometer was utilized for tilt angle measurement, and the fuzzy

control algorithm was programmed on DSPIC microcontroller. The DSPIC is 16-bit controller, modified Harvard architecture, C-complier optimized, 30 MIPS operation, motor control PWM module, 10-bit ADC with 1 MSPS conversion rate, and single-cycle multiple-add operation. The authors stated that good performance was achieved with the embedded fuzzy controller.

Chen et al. (2011) have built a two wheeled scooter similar to Segway and used brushless DC motors. They have designed a fuzzy logic based controller for balancing the scooter. They used the same hardware in (Jean and Wang 2009), the MCU and the CPLD, for real time implementation of 49 fuzzy rules. They used a double axis accelerometer and the CRS03-02 gyroscope to measure the tilt angle. The authors have not explained the implementation of the position controller.

Lee and Jung (2012) have used a low cost sensor to detect the tilt angle for a two wheeled robot called 'BalBOT'. They presented a filtering algorithm by combining a complementary filter with a Kalman filter to achieve accurate measurements. The robot was controlled by simple PD and PID controllers to control balancing angle and tracking of position and heading angle. Real-time control was achieved by embedding the control and filtering algorithm onto a DSP board. Wireless communication was used to control the movements of the robot by a joystick remotely. The authors stated based on the experimental results show that the MIPS is robust against external disturbances such as intentional hits, and tracked the desired trajectories generated by the joystick while balancing.

Farooq et al. (2012) have designed a fuzzy logic controller for balancing a two wheeled robot by utilizing the tilt angle measurements to generate the controller action. The AT89C52 microcontroller was used for real time implementation of the controller designed and optimized using the fuzzy logic toolbox of MATLAB. The authors reported that the robot remained in its inverted position even when a disturbance is applied.

Rukidi et al. (2014) utilized the ARM Cortex-M4 processor of the STM32F3-Discovery development board in order to implement the controller of a practical teaching two-wheeled robot platform. The presented controller comprised classical PID for tilt angle and position control of the robot. The STM32F3-Discovery

development board was used to implement the sensor fusion technique represented by a complementary filter for tilt angle measurement. The reported experimental results showed that the two-wheeled robot with the ARM Cortex-M4 processor provides the student a robust educational platform that can exhibit many aspects of control theory and facilitate the real time implementation without processing limitation.

1.2.3 Body design

All two wheeled robots or vehicles share the same basic design which is based on the inverted pendulum concept. They are constructed from two coaxial wheels, which are used for driving and balancing with small auxiliary casters in some designs for safety reasons or for resting when the controller is switched off. However, in the literature one could identify some differences in the two wheeled robot designs and functionality.

Some of the two wheeled robot have been designed as transporter systems. Such systems have gained great interest in research and this has led to commercial products such as Segway Human Transporter (Kamen 2001) constructed as a standing base and handlebar as shown in Figure 1.2. It is driven by the driver by leaning the handlebar forward and backward with a steering mechanism which is mounted on the handlebar. Similar studies and analyses are found in (Solis et al. 2009), (Tsai et al. 2010), (Chen et al. 2011), (Li et al. 2011), and (Hata and Takimoto 2014).

Figure 1.2: ***Segway Human transporter (http://www.segway.com)***

In the meantime, development and studies have been conducted on the two wheeled transporter systems that have seats such as (Baloh and Parent 2003), (Hyung et al. 2010), (Petrov and Parent 2010), (Li and Yang 2011) and (Vermeiren et al. 2011). Such research lead to collaboration between GM motors and Segway to build a two-seater vehicle based on two wheeled inverted pendulum system named Personal Urban Mobility and Accessibility PUMA as in Figure 1.3.

Figure 1.3: ***Personal urban mobility and accessibility (PUMA) (http://www.gizmag.com/segway-car-project-puma/11413/)***

A series of studies by Takahashi et al. (1999) were conducted to design and control a motorized wheel chair. They designed a mechanism for the wheelchair to be able to raise the front wheels while maintaining balance posture on the rear two wheels. Such mechanism allows the wheel chair to climb steps and move forward and backward while balancing, depending on the human body inclination and posture. The motors in the chair were controlled by a simple proportional integral (PI) controller in order to stabilize the wheel chair. Further work and analysis were conducted by the authors on wheel-raising and balancing of the robotic wheelchair in (Takahashi et al. 2001), (Takahashi et al. 2003) and (Takahashi and Kohda 2005). Such research was translated into the iBot a commercial robotic wheelchair shown in Figure1.4, which has the ability to balance on two wheels and climb stairs (Kamen 2001).

Figure 1.4: ***The independence technology iBot wheelchair (http://www.huey091foundation.org/)***

Salerno and Angeles (2007) have classified the two wheeled robots into three categories based on the balancing mechanism. Firstly, robots without any stabilization of the body. Such robots usually have a large wheels and their body may rock back and forth as only two points touch the ground. Secondly, robots with passive mechanical stabilization of the body. Such robots have a mechanical stabilizer such as a handle attached to the robot manoeuvred by the user and a sliding supporting point. However, when these designs are working on inclined surfaces, they do not behave as good as on flat surfaces. The third class is robots with active stabilization of the body, where several sensors, actuators, and subsystems are used achieve body stabilization. Based on this classification the authors proposed a new model of the two wheeled robot and analysed the mathematical model for the controllability issue.

Murakami and Sasaki (2009) have developed a two wheeled robot equipped with four links manipulator that is mounted on the base platform to realize a pushing operation as in Figure 1.5. They considered the sagittal motion of mobile manipulator only in the system model. They developed the control law to realize the end-effector force control and pushing operation of mobile manipulator. The stabilization of the base body was achieved by controlling the tilt angle of the body and the centre of gravity (COG) of the manipulator itself. The authors stated that simulation results were verified with an experimental test which showed a stable operation of the robot. Similar structural design was studied and presented in (Abeygunawardhana and Toshiyuki 2007) and (Abeygunawardhana and Toshiyuki 2008).

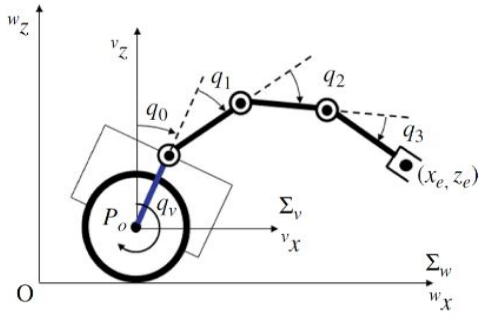


Figure 1.5: ***Two-wheel inverted pendulum mobile manipulator (Murakami and Sasaki 2009)***

Two wheeled robots equipped with arms have also been reported in the literature. Various purposes of the attached arms have been highlighted such as in (Seong Hee and Takahashi 2007), (Kuindersma et al. 2009) and (Feng et al. 2011), shown in Figure 1.6, where the authors have presented many of the characteristics of a humanoid robot and utilized the arms to support stand to set motion and posture balancing control. Other inverted pendulum based robots equipped with arms such as that in (Fukushima et al. 2015) were designed to support the transformation of the robot form four wheels into two-wheeled mode that has advantages of high viewing position and small turning radius.

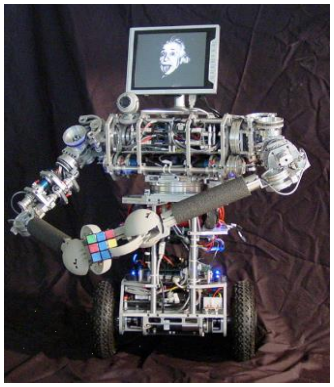


Figure 1.6: ***The ubot5 two-wheeled robot with arms (Kuindersma et al. 2009)***

Nakagawa et al. (2011) have proposed the two-wheeled inverted pendulum vehicle derived by human pedalling force. The vehicle consists of the stabilization control by the in-wheel DC motors and the driving power by a human. The authors designed the vehicle to have two driving modes. First mechanical, as the force of

human pedalling was transmitted to the each wheel mechanically while the stabilization control was done by controlling the motor torque to the both wheels based on the posture of the vehicle. Second electrical, as the pedalling force was used to charge the batteries and the DC motors are used for driving and stabilizing the vehicle. Figure 1.7 shows the prototype of the vehicle.

Figure 1.7: ***The two-wheeled inverted pendulum vehicle with pedals (Nakagawa et al. 2011)***

Goher and Tokhi (2010) have designed and modelled a new structure of a two wheeled robot with an additional degree of freedom along its intermediate body, vertical axis, to provide a payload with the ability to reach different heights as illustrated in Figure 1.8. Another pace of work was conducted in (Goher et al. 2010) which apply the raising of the payload along the vertical axis mechanism to a double inverted pendulum system. Moreover, Almeshal et al. (2011) have presented the modelling and controlling of a two wheeled robotic wheelchair, based on the double inverted pendulum concept, with the ability to lift the seat up to demanded height.

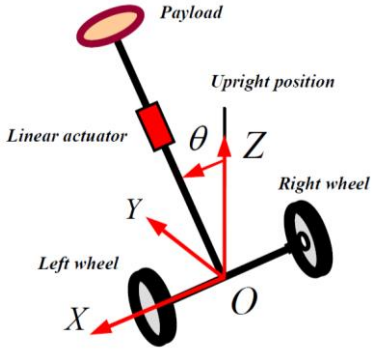


Figure 1.8: ***The two-wheeled inverted pendulum vehicle with extended rod (Goher and Tokhi, 2010)***

1.3 Objectives of the research

Two wheeled inverted pendulum vehicle is considered to be highly coupled, non-linear and under actuated system. Such systems are considered to be utilized for indoor and outdoor environmental usage where many unexpected conditions could apply to the vehicle and affect its performance. The associated uncertainties need to be considered when designing control strategies for such machines.

Previous studies on two wheeled inverted pendulum based robots have revealed many important modelling and control schemes and the interest of researchers. The majority of the covered literature concentrated on developing the control algorithms required to keep the two-wheeled inverted pendulum robot in a balancing state, while other works discussed the dynamics of the system and the methods to realize such system in real time. Most of these contributions assumed 3-DOF systems with either load-free model or fixed load size and position model of the robot, where the load is considered to be concentrated at the centre of mass (COM) of the system. This observation raises a question that what is the potential of having a dynamic payload on top of the vehicle and how to control it. In addition, what is the possibility of having a dynamic movable payload that would allow the payload to be lifted to a demanded height. Such issues were analysed numerically using the Simulink of MATLAB in (Goher and Tokhi 2010) and (Goher et al. 2010).

In the literature, researchers have chosen various techniques to implement the two wheeled robotic system in real-time. Some of them have presented methods to achieve accurate sensor measurements of the system in order to replace the expensive IMU with cost efficient sensors. Others have discussed real-time implementation of the control system aspects and hardware which is used to build the system. Most of the reviewed two wheeled systems have been implemented in real-time using either the real-time workbench of MATLAB or development boards such as DSP, FPGA, MUC. Whereas, others have realized their controller using laptops or mini PC boarded on the robot itself as in (Coelho et al. 2008), and few researchers have chosen single board computers (SBC) in real time implementation of the two wheeled robot system.

Few works presented on two wheeled inverted pendulum system moving on an inclined surface. Despite previous studies on two wheeled inverted pendulum based robotic vehicles, there is still need for mobile inverted pendulum vehicles that are capable of moving in different environments and can cope with different types of surfaces such as an inclined surface. Relevant designs, such as the iBot by the Independence Technology Company, have had production discontinued because of a high selling price. With continued demand from disabled and elderly people for this type of product, new designs should be within an affordable production and selling price range to target a large sector of the population.

This research is intended to validate practically a novel two wheeled robot based on the inverted pendulum concept with an extended height and moving payload that has the ability to manoeuvre on flat and inclined surfaces. The model will form a basis for more applications to be developed, such as advanced human transporters and wheelchairs for elderly and disabled persons. These types of robots that reach extended heights can help a person with daily activities such as reaching higher shelves, in addition to moving on inclined surfaces.

One of the challenging points is to consider various positions of the load by studying the system under the condition of changing the level of the load. Another interesting point is the resulting impact on system behaviour due to changing the position of that load. This is supposed to be carried out in the same instance with changing the size of the applied load to discuss the suitability of the selected control algorithm to keep the whole system in a balancing situation. Specially, some phenomena are expected to act as noise on the system and affect the performance of the control algorithm due to the movement of the applied load. The main objectives of this research are outlined as follows:

- Study of the mechanisms of two wheeled robots and the problems associated with their control and investigate development of control algorithms for two wheeled inverted pendulum robots.
- Development of control algorithms for a novel two wheeled robot based on the inverted pendulum concept with an extended height and moving payload.

- Investigate and resolve issues of stability, balance, and sensor fusion techniques to determine the motion state of the two wheeled robot platforms.
- Implement the developed control algorithms on an embedded real-time system platform, and test and evaluate performance of developed control approaches in different robot motion scenarios.

1.4 Thesis outline

Chapter 1: This chapter introduces the two wheels robot system which is based on the inverted pendulum concept. Furthermore, it describes its benefits, associated devices and control strategies. Moreover, this chapter provides an introduction to the control strategies, real-time implementation techniques that been used with the two wheels robot systems.

Chapter 2: This chapter describes the hardware of the two wheels robot and the integration of its various parts. Also introduces the mathematical modelling and the integration of this with Matlab / Simulink, for use as a platform for analysis, design and evaluation of the developed strategies and approaches in this research.

Chapter 3: This chapter devoted to the sensing techniques of the two wheels robot system states, position and tilt angle, with the associated sensors. Furthermore, it introduces the signal conditioning and filtering methods such as the commentary filter and the Kalman filter.

Chapter 4: This chapter demonstrates the development of different classic control structures, PID and hybrid PID, of the two wheeled robot. Two different controller structures are developed and used. Also, it depicts the experimental results of the two wheels system under different operating conditions such as fixed payload position, movable payload, different payload weight, and balancing on an inclined surface.

Chapter 5: In this chapter two fuzzy logic control approaches are investigated, a PD-like fuzzy control strategy has been adopted to replace the tilt angle controller of the two wheels robot. Furthermore, it describes a new coding technique of the fuzzy logic controllers. This technique has been developed for the real-time implementation of the controller. The results show that fuzzy logic controller has better performance than the PID controller in terms of reduce the tilt angle oscillation and the energy consumption.

Chapter 6: This chapter summarizes the main conclusions of the work and makes recommendations for further work

1.5 Contributions

Flexible and efficient mobile robots are becoming important in the industry to be used as transport, operation, and exploration. In addition, they are useful for exploring hostile environments for carrying out dangerous tasks such as hailing radioactive waste, decontaminating nuclear reactors, and sweeping mine fields. Furthermore, mobile robots can serve as test platforms for a range of experiments either in academia for applying and developing new algorithms, or in sensing the environmental changes to then respond to with intelligent choices. The main contributions of this research are outlined as follows:

- Validate experimentally a novel configuration of two wheeled inverted pendulum vehicle with additional degrees of freedom realize by a movable payload. This can serve as new platform for further academic research.
- Design and real time implementation of PID based control system to maintain the balancing state of the robot using compromised cost efficient hardware. Furthermore, a PC environment in Microsoft visual basic has been developed to offer a simple GUI with the robot. This provides a real-time monitoring and data logging and tuning of the robot's parameters as well as a remote connection with the robot to send navigation commands.
- Development of a new binary coding technique to implement fuzzy controllers on microcontrollers' hardware. Such coding facilitates the use of microcontrollers with small memory space. Furthermore, a PC software has

been designed and built to help researchers/users in the design, editing and coding of fuzzy controllers.

- Design and realisation of an intelligent controller for the two-wheeled robot with by PD-like fuzzy control. The controller has been binary codded and implemented on the microcontroller development board for real-time execution.

1.6 Publication

➢ Peer reviewed journal papers

- S. C. Abdulla, **Sayidmarie, O.K.,** and M. O. Tokhi, "Functional electrical stimulation-based cycling assisted by flywheel and electrical clutch mechanism: A feasibility simulation study," *Robotics and Autonomous Systems,* vol. 62, pp. 188-199, 2014.

➢ Conference publications

- **Sayidmarie, O.K.,** Tokhi, M. O., and Agouri, S. A., Impact of dynamically moving payload on two wheeled robot stability. *The 19th International Conference on Methods and Models in Automation and Robotics (MMAR 2014)*, Miedzyzdroje, Poland, 2 - 5 September 2014.
- **Sayidmarie, O.K.,** Tokhi, M. O., and Agouri, S. A., Real-time validation of a novel two-wheeled robot with a dynamically moving payload. *The 23rd IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN 2014),* Edinburgh, Scotland, UK, August 25-29, 2014.
- **Sayidmarie, O.K.,** Tokhi, M. O., and Agouri, S. A., Balancing and control of a two-wheeled robot on inclined surface. *The 17th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR 2014),* Poznan University of Technology, Poznan, Poland, 21 - 23 July 2014.

- Agouri, S. A., Tokhi, M. O., Al-Meshal, A. M., and **Sayidmarie, O.K.,** Performance of a two wheeled robot with extendable intermediate body on irregular terrains, *The 17th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR 2014)*, Poznan University of Technology, Poznan, Poland, 21 - 23 July 2014.
- Nasir, A. N. K., Tokhi, M. O., **Sayidmarie, O.K.,** and Ismail, R. M. T. R., A novel adaptive spiral dynamic algorithm for global optimization. *The 13th UK workshop on computational intelligence* (UKCI 2013), 9-11 September 2013.
- Agouri, S. A., Tokhi, O., Almeshal, A., **Sayidmarie, O.K.,** Goher, K. M. & Muscat, O. Modelling and control of two-wheeled vehicle with extendable intermediate body on an inclined surface. *The 12th IASTED international conference on artificial intelligence and applications*, Innsbruck, Austria, February 11 – 13, 2013.
- **Sayidmarie, O.K.,** Tokhi, M. O., Agouri, S. A., Al-Meshal, A. M., and Goher, K. M., Design And Real Time Implementation of A Fuzzy Logic Control System For A Two-Wheeled Robot. *The 17th International Conference on Methods and Models in Automation and Robotics (MMAR 2012)*, Miedzyzdroje, Poland, 27 - 30 August 2012.
- Abdulla, S. C., **Sayidmarie, O.K.,** Gharoni, S., and Tokhi, M. O. Modelling and Control of a Novel FES Driven Assisted Cycling Mechanism. *The 17th International Conference on Methods and Models in Automation and Robotics (MMAR 2012)*, Miedzyzdroje, Poland, 27 - 30 August 2012.
- Agouri, S. A., Tokhi, M. O., Al-Meshal, A. M., **Sayidmarie, O.K.,** and Goher, K. M., Dynamic Modelling of a New Configuration of Two Wheeled Robotic Machine on an Inclined Surface. *The 17th International Conference on Methods and Models in Automation and Robotics (MMAR 2012)*, Miedzyzdroje, Poland, 27 - 30 August 2012.
- Short, A. R., **Sayidmarie, O.K.**, Agouri, S. A., Tokhi, M. O., Goher, K. M., and Al-Meshal, A. M., Real Time PID Control of A Two-Wheeled Robot. *The 15th International Conference on Climbing and Walking Robots and the*

Support Technologies for Mobile Machines (CLAWAR 2012), Johns Hopkins University, USA, 23 - 26 July 2012.

- Al-Meshal, A. M., Goher, K. M., Tokhi, M. O., **Sayidmarie, O.K.,** and Agouri, S. A., Hybrid Fuzzy Logic Control Approach of A Two Wheeled Double Inverted Pendulum Like Robotic Vehicle. *The 15th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR 2012)*, Johns Hopkins University, USA, 23 - 26 July 2012.
- Goher, K. M., Tokhi, M. O., Al-Meshal, A. M., **Sayidmarie, O.K.,** and Agouri, S. A., Impact of Payload Inertia on the System Damping Characteristics of a Two-Wheeled Robotic Machine. *The 15th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR 2012)*, Johns Hopkins University, USA, 23 - 26 July 2012.

CHAPTER 2

Hardware and software description of the two wheeled robot

2.1 Introduction

Designing a balancing two-wheeled platform consists of using several hardware components including microcontroller board, sensors, motors and physical structure of the two-wheeled robot chassis. Also, programming and testing the code in the device involves the use of different software packages. This chapter covers these two main aspects of the two-wheeled platform, the hardware description and the software description, and how they are integrated with each other to show a working two-wheeled robot.

2.2 Robot's hardware description

A prototype of the two-wheeled robot, shown in Figure 2.1, has been designed and developed in the Automatic Control and Systems Engineering department. This will be used to implement, develop and test the control algorithm of the robot.

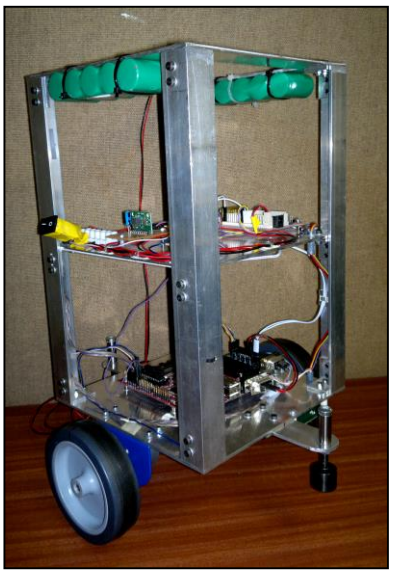


Figure 2.1: ***Two wheeled robot test rig prototype***

2.2.1 Mechanical parts

In this section, a description of the main mechanical parts of the two-wheeled robot test rig is presented. These parts include the chassis of the robot, wheels and motors, and the payload actuation unit.

The chassis of the robot is constructed to be rigid but light at the same time so Aluminium sections were used. The chassis has dimensions (20 * 20 * 31 cm), and consists of equally divided three layers.

Two DC motors with a dual H-Bridge DC motor controller board, which supply the force to drive the robots motion, are attached to the bottom layer.

The middle layer of the chassis houses two main units. The first is a microcontroller development board, while, the second unit is an Inertial Measurement Unit (IMU). The IMU consists of a low g accelerometer and a gyroscope, which is used to measure the posture of the robot.

The upper layer of the chassis houses a Bluetooth model, which is used to communicate with the client PC. In addition, the batteries pack is placed on this layer in order to shift up the centre of gravity of the robot.

2.2.2 Electrical and electronic parts

In order to make the two-wheeled robot rig functional, various electrical parts and electronic circuits boards were utilized in the design such as power supply, sensors, microcontroller board, motor drivers, and wireless data links; refer to appendix A for more specification.

Figure 2.2 describes the interconnection and data flow paths between the main circuits and parts of the robot. The microcontroller oversees all operating aspects of the two-wheeled robot. The MCU is an 8-bit microcontroller runs at clock frequency of 16MHz. It has digital I/O, analogue I/O and serial ports for controlling various circuits. For example, the RX, TX lines are used for the RS-232 interface and the I^2C bus interface.

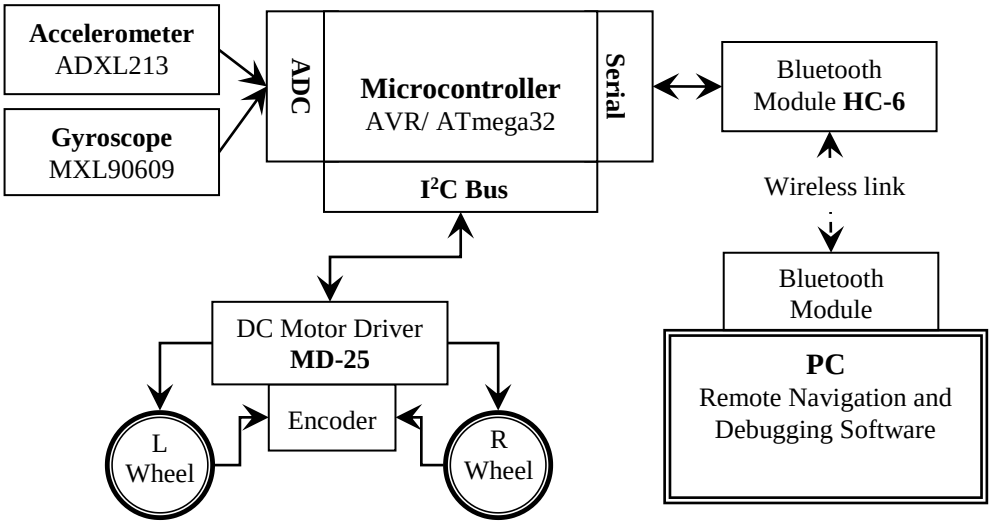


Figure 2.2: ***Data flow diagram of the two-wheeled robot***

Both of the tilt angle sensors; the accelerometer and the gyroscope; are connected to the analogue I/O lines of the microcontroller. While the motors deriver board MD-25 is connected the microcontroller via the I^2C bus. This board reads the wheel's encoders of the robot and controls the rotation speed and direction of the wheels.

Finally, a serial to Bluetooth module is connected to the microcontroller board in order to provide a wireless link with a PC that runs the remote navigation and debugging software.

2.3 Robot's mathematical modelling

In this section, a simplified mathematical model of the two-wheeled robot is developed with the assumption that the robot is moving on flat regular surfaces. This model will be the basic model of the robot with a specific payload weight and at a fixed position. In subsequent sections, the mathematical model will be developed with the inclusion of variable payload weight and position.

Generally, the model comprises a set of dynamic equations describing the interaction between the motion of the system and the forces or torques. These equations are essential for designing the system parts, choosing the right

motor's/drive's power, and building a simulation model for the system. Such model will be important in designing and developing control strategies for the system.

Mainly, based on the mechanical system constraints there are two common techniques for developing the equations of the system; Newton-Euler equations and Lagrange`s formulation. Newton's equations treat each rigid body separately and model explicitly the constraints through the forces required to enforce them. Whereas Lagrange and d'Alembert offer procedures to eliminate the constraints from the dynamic equations leading to simpler equations. In the literature of robotics, Lagrange's formalism is often the method of choice, as joints and other mechanical components in the robots, which create constraints constitute, are one of the defining features of robots.

Rather than using the Newton-Euler formulation to derive the system dynamics, the Euler-Lagrange approach is utilized for modelling the mechanical parts of the robot, as naturally, two-wheeled robots are non-linear, and coupled systems. In this approach both of the potential and kinetic energies of the system are utilized to express the dynamics of the two wheeled robot. Hence, the variables are scalar quantities and the formulation is simple compared to Newton-Euler.

2.3.1 TWR with a fixed payload

Figures 2.3 and 2.4 show an illustration of the two-wheeled robot test rig, where the corresponding parameters of the robot are listed in Table 2.1. The robot consists of the two main parts: the chassis of the robot which represents the inverted pendulum intermediate body. While the second part of the robot is represented by two driving wheels attached to DC motors. These motors act independently of each other in order to give the robot the ability to manoeuvre. However, in this model both motors will be considered to be connected to the same control signal. Hence, forward/ backward linear motion of the robot can be examined.

A couple of assumptions are considered during derivation of the mathematical model. The wheels of the robot are in full contact with ground at all times, all frictions are neglected in the calculation, and the robot is constructed from a rigid parts with a homogeneous mass distribution.

The robot's dynamic model can be divided into two separated sub-models. The first sub-model is a non-linear model of the robot's chassis which is more or less similar to an inverted pendulum over cart model. The second sub-model is a linear model of the DC motors that drives the wheels of the two-wheeled robot.

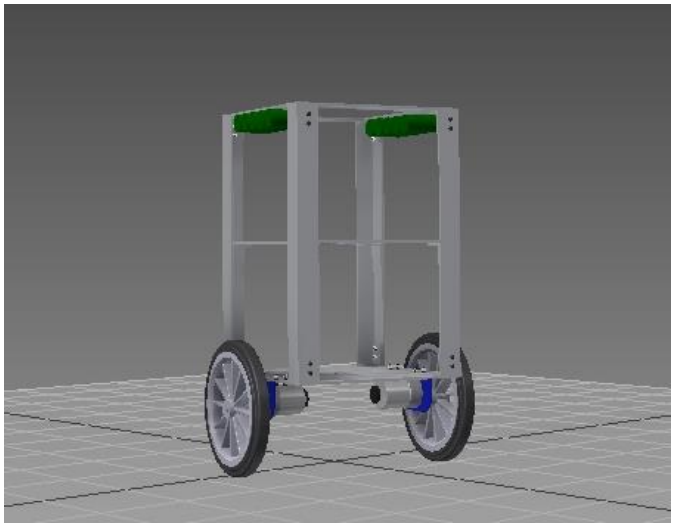


Figure 2.3: ***CAD model of the two-wheeled robot***

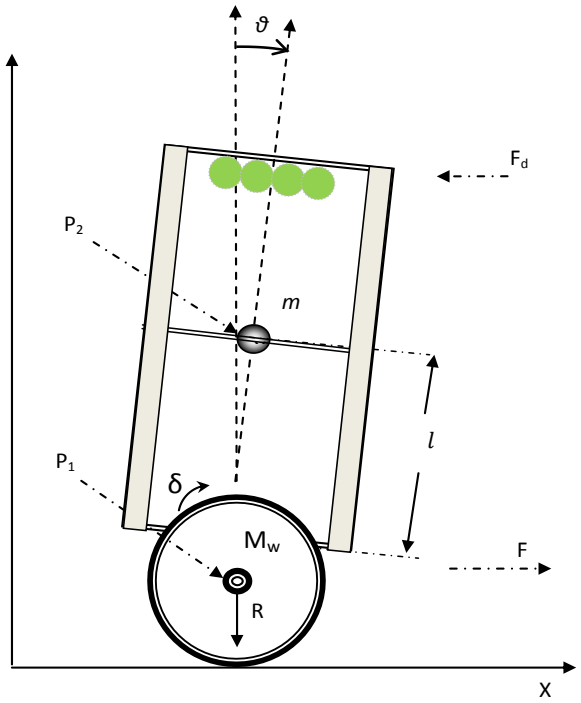


Figure 2.4: ***Two-wheeled robot structure diagram***

Table 2.1: ***Nomenclature of the robot***

Variable	Description	Unit
F	Linear driving force	N
F_d	External disturbance force	N
J_I	Mass moment of inertia of the chassis	kg.m^2
J_w	Mass moment of inertia of the wheel	kg.m^2
l	Distance to the centre of mass	m
M	Mass of the payload	kg
m	Mass of the chassis	kg
M_w	Mass of the wheel	kg
R	Radius of the wheel	m
δ	Angular displacement of the wheel	rad/sec
θ	Tilt angle of the chassis	rad

2.3.1.1 Mathematical model of the TWR chassis

Lagrange dynamic formulation is used to obtain equations of motion for the two-wheeled robot system. The dynamic equations of the system can be expressed for *n* degrees of freedom (DOF) based on kinetic energy and potential energy of the system as follows

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i \quad (2.1)$$

$$L = T - V \quad (2.2)$$

where,

L, is the Lagrangian function,

Q_i is the generalized force associated with a generalized coordinate q_i,

q_i is the generalized coordinate,

i is the i^{th} degree of freedom of the system,

T is the total kinetic energy of the system, and

V is the total potential energy of the system.

The two-wheeled robot system will have two degrees of freedom (n=2); which are linear displacement of the robot δ and tilt angle of the robot θ. Hence, the force and coordinate matrices are defined as

$$q = [\ \delta\ \theta\ ] \quad (2.3)$$

$$Q = [F\ \ F_d] \quad (2.4)$$

2.3.1.1.1 Kinetic energy calculation

Generally, the kinetic energy of the two wheeled robot system is the sum of the translation energy T_t and the rotation energy T_R of the different parts of the robot as expressed in equations below.

$$T = T_t + T_R \quad (2.5)$$

$$T_t = \frac{1}{2} M\ v^2 \quad (2.6)$$

$$T_R = \frac{1}{2} Jw^2 \quad (2.7)$$

where,

T_t is the transitional kinetic energy,

M is the mass,

v is the transitional, linear, velocity,

T_R is the rotational kinetic energy,

J is the moment of inertia, and

w is the angular velocity.

Similar to the inverted pendulum over a cart system, the two wheeled robot system has two main parts. The first part includes the motors, gearbox and wheels and it replicates the cart. The second part is the chassis which replicates the intermediate body. In order to calculate the kinetic energy of the robot, the velocity of each part of the robot should be evaluated first. Referring to Figure 2.4 the velocity of the cart can be expressed as:

$$\overrightarrow{V_{p1}} = \frac{d}{dt}x = \dot{x} = R\dot{\delta} \quad (2.8)$$

$$|V_{p1}| = R\dot{\delta} \quad (2.9)$$

where,

x is the travelled distance in the x axis,

R is the radius of the wheels, and

$\dot{\delta}$ is the angular velocity of the wheel.

Similarly, the velocity of the chassis can be expressed by taking the velocity of the centre of mass point in terms of the tilt angle and the cart velocity as

$$\overrightarrow{V_{p2}} = \frac{d}{dt}\left(x - lsin(\theta)\right)_x + \frac{d}{dt}(lcos(\theta))_y \quad (2.10)$$

$$|V_{p2}| = \sqrt{\left(\dot{x} + l\dot{\theta}\cos(\theta)\right)^2 + \left(-l\dot{\theta}\sin(\theta)\right)^2} \quad (2.11)$$

where,

x is the travelled distance in the x axis,

l is the distance of the centre of mass of the chassis, and

θ is the tilt angle of the chassis.

Then the transitional kinetic energy of the robot can be calculated as

$$T_t = \frac{1}{2}M_w\,{V_{p1}}^2 + \frac{1}{2}m_I{V_{p2}}^2 \quad (2.12)$$

where,

M_w is the mass of the wheels, and

m_I is the total mass of the chassis and the payload.

Substitute Equations (2.9) and (2.11) into (2.12) yield

$$T_t = \frac{1}{2} M_w \, (\dot{x})^2 + \frac{1}{2} m_I \left(\left(\dot{x} + l\dot{\theta} \cos(\theta) \right)^2 + \left(-l\dot{\theta} \sin(\theta) \right)^2 \right) \quad (2.13)$$

$$T_t = \frac{1}{2} M_w \left(R\dot{\delta} \right)^2 + \frac{1}{2} m_I \left(R^2\dot{\delta}^2 - 2Rl\dot{\delta}\dot{\theta} \cos(\theta) + l^2\dot{\theta}^2 \right) \quad (2.14)$$

The rotational kinetic energy of the main parts of the robot, the wheels and the chassis, can be expressed as

$$T_R = \frac{1}{2} J_w \dot{\delta}^2 + \frac{1}{2} J_I \dot{\theta}^2 \quad (2.15)$$

where,

J_w is the moment of inertia of the wheel,

$\dot{\delta}$ is the angular velocity of the wheels,

J_I is the moment of inertia of the chassis, and

$\dot{\theta}$ is the angular velocity of the chassis.

Thus the total kinetic energy of the two-wheeled robot can be expressed by substituting Equations (2.13) and (2.14) into (2.5) yielding

$$T = \frac{1}{2} \Big[M_w \left(R\dot{\delta} \right)^2 + m_I \left(R^2\dot{\delta}^2 - 2Rl\dot{\delta}\dot{\theta} \cos(\theta) + l^2\dot{\theta}^2 \right) + J_w\dot{\delta}^2 + J_I\dot{\theta}^2 \Big] \quad (2.16)$$

2.3.1.1.2 Potential energy calculation

Similar to kinetic energy calculation, the total potential energy of the two-wheeled robot is the summation of all parts energies, the wheels and the chassis. Since the robot is assumed to remain in full contact with the ground and moving on flat surfaces, there is no potential energy component in the Z direction. Hence, the potential energy of system's parts can be expressed as

$$V = M_w g R + m_I g (R + l\cos(\theta)) \quad (2.17)$$

where,

V is the total potential energy of the robot,

M_w is the mass of the wheel,

m_I is the mass of the chassis,

R is the radius of the wheel,

l is the distance of the centre of mass of the chassis, and

θ is the tilt angle of the robot.

2.3.1.1.3 Lagrangian function calculation

By substitution of the kinetic and the potential energy equations of the two-wheeled robot Equations (2.16) and (2.17) into Equation (2.2), the Lagrangian function can be expressed as

$$L = \left[\frac{1}{2}(M_w + m_I)R^2\dot{\delta}^2 - m_I lR\dot{\delta}\dot{\theta}cos(\theta) + \frac{1}{2}\left(m_I l^2\dot{\theta}^2 + J_w\dot{\delta}^2 + J_I\dot{\theta}^2\right)\right] - [M_w gR + m_I g(R + lcos(\theta))] \quad (2.18)$$

Hence, the dynamic equation of the driving force F of the robot can be calculated by solving the Lagrangian formulation in Equation (2.1) as

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\delta}}\right) - \frac{\partial L}{\partial \delta} = F \quad (2.19)$$

$$\frac{\partial L}{\partial \dot{\delta}} = (M_w + m_I)R^2\dot{\delta} - m_I lR\dot{\delta}\dot{\theta}cos(\theta) + J_w\dot{\delta} \quad (2.20)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\delta}}\right) = (M_w + m_I)R^2\ddot{\delta} - m_I lR\ddot{\theta}cos(\theta) + m_I lR\dot{\theta}^2 sin(\theta) + J_w\ddot{\delta} \quad (2.21)$$

$$\frac{\partial L}{\partial \delta} = 0 \quad (2.22)$$

Substituting Equations (2.20) and (2.22) into (2.19) yield

$$F = (M_w + m_I)R^2\ddot{\delta} - m_I R l\ddot{\theta}\, cos(\theta) + m_I R l\dot{\theta}^2\, sin(\theta) + J_w\ddot{\delta} \quad (2.23)$$

The second dynamic equation of the robot which represents the disturbance forces on the TWR chassis F_d can be calculated by solving the Lagrangian formulation for the second coordinate as

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = F_d \quad (2.24)$$

$$\frac{\partial L}{\partial \dot{\theta}} = -m_I l R\dot{\delta} cos(\theta) + m_I l^2\dot{\theta} + J_I\dot{\theta} \quad (2.25)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = -m_I l R\left(\ddot{\delta} cos(\theta) - \dot{\delta}\dot{\theta} cos(\theta)\right) + m_I l^2\ddot{\theta} + J_I\ddot{\theta} \quad (2.26)$$

$$\frac{\partial L}{\partial \theta} = m_I l R\dot{\delta}\dot{\theta} sin(\theta) + m_I l g sin(\theta) \quad (2.27)$$

Substituting Equations (2.26) and (2.27) into (2.24) yield

$$F_d = m_I l^2\ddot{\theta} + J_I\ddot{\theta} - m_I l g sin(\theta) - m_I l R\ddot{\delta} cos(\theta) + f_d \quad (2.28)$$

where f_d is the total external forces on the TWR chassis

2.3.1.2 Mathematical model of the motors

The two-wheeled robot is driven by two geared head permanent magnet DC motors that generate the required driving force *F* of the robot. In order to relate the applied voltage with the torque of the motor's shaft, two dynamic equations can be developed based on the electrical and mechanical characteristics of the motor. Figure 2.5 shows the electromechanical structure of a standard DC motor.

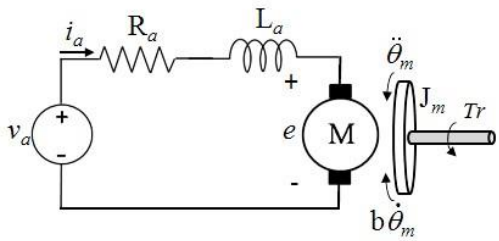


Figure 2.5: ***A DC motor model diagram***

Using Kirchhoff's law, the summation of the voltages in a circuit equal to zero, the following equation can be obtained

$$v_a = i_a R_a + L_a \frac{di_a}{dt} + e \quad (2.29)$$

$$e = K_v * \dot{\theta}_m \quad (2.30)$$

where,

v_a is the applied voltage,

i_a is the current of the motor,

R_a is the resistance of the motor,

L_a is the inductance of the motor,

e is the voltage of the back electromotive force *emf*,

K_v is the velocity constant, and

$\dot{\theta}_m$ is the angular velocity of the motor.

Based on the mechanical characteristic of the motor and Newton's law of the sum of the torques of the motor must equal zero; the following equation is obtained

$$Tr = K_m i_a - b\dot{\theta}_m - J_m \ddot{\theta}_m \quad (2.31)$$

where,

Tr is the shaft torque,

K_m is the motor (torque) constant,

i_a is the current of the motor,

J_m is the moment of inertia of the motor,

$\dot{\theta}_m$ is the angular velocity of the motor,

b is the friction constant, and

$\ddot{\theta}_m$ is the angular acceleration of the motor.

Substituting for e from Equation (2.30) into Equation (2.29) and rearranging the equations in terms of the highest derivative yield the dynamic equations of the motor as

$$\ddot{\theta}_m = \frac{i_a - \dot{\theta}_m}{J_m} K_m - \frac{Tr}{J_m} \tag{2.32}$$

$$\frac{di_a}{dt} = \frac{R_a}{L_a} i_a + \frac{K_v}{L_a} \dot{\theta}_m + \frac{V_a}{L_a} \tag{2.33}$$

In practice, the torque generated by the DC motor shaft with any gear assembly is converted to the required force to drive the robot. This torque is related to the driving force of the robot as

$$F = N\ Tr\ R \tag{2.34}$$

where,

N is the gear ratio,

Tr is the shaft torque of the motor, and

R is the radius of the wheel.

2.3.2 TWR with moving payload

The proposed movement of the payload was achieved by means of a linear actuator. The actuator is considered to activate the payload in order to move along the intermediate body, the vertical axis of the chassis of the two-wheeled robot, with prescribed motion characteristics. Therefore, a mechanical actuation unit was considered to link the actuator of the payload with the chassis of the two-wheeled robot. The payload actuation unit design will be described in the next section. The actuator with other supportive mechanical parts were placed at the centre of the middle layer of the two-wheeled robot. Hence, the payload can be lifted by approximately 20 cm from the original COM of the robot. Also, hardware modifications and improvements of the two-wheeled robot design are applied in order to overcome some of the design issues and enhance the performance. These improvements involve a replacement of the wheels, motors, tilt sensors, and the main microcontroller board Figure 2.6 shows the actual hardware of the two-wheeled robot rig after adding the payload actuation unit. Where Figure 2.7 explains the data flow diagram.

a. Before ***b. After***

Figure 2.6: ***The modified two-wheeled robot test rig***

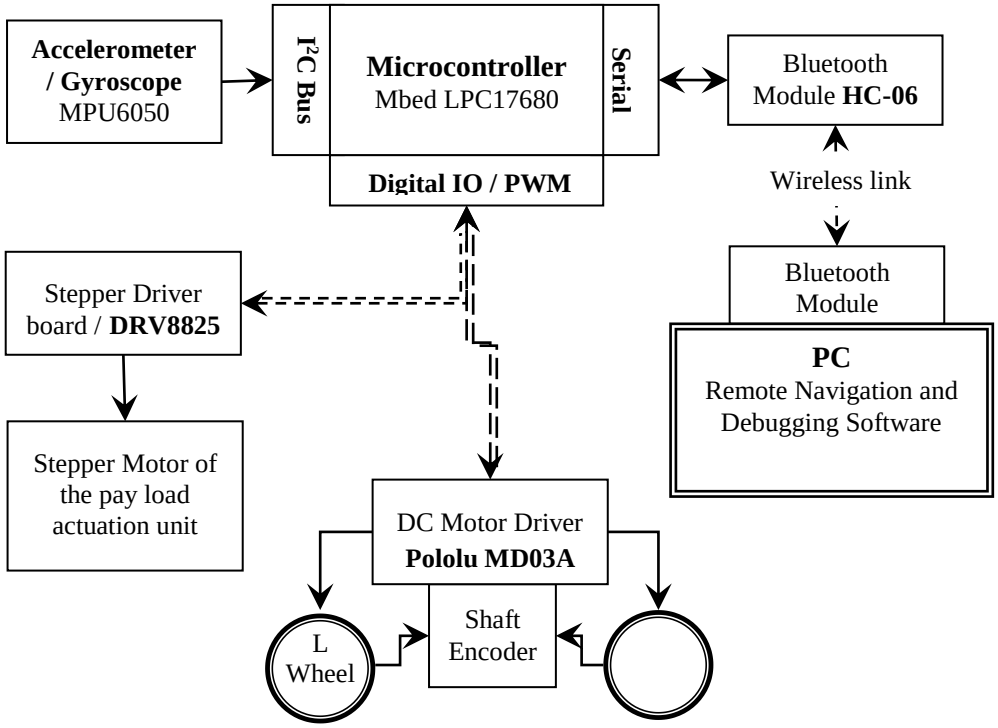


Figure 2.7: ***Data flow diagram of the two-wheeled robot with movable payload***

The microcontroller board of the two-wheeled robot was replaced with the Mbed development board to enhance the proceeding speed. The Mbed LPC17680 board is a 32-bit ARM microcontroller runs at clock frequency of 96MHz. It provides several interfaces such as serial ports, up to 3 ports, Ethernet, USB device/host, CAN, two SPI buses in addition to analogue and digital I/O lines.

Both of the tilt angle sensors; the accelerometer and the gyroscope; are replaced with a single module MPU6050, which is 3-axis gyroscope / accelerometer. The MPU6050 sensor offers digital factory calibrated measurements over I^2C interface. While the motors deriver board MD03A and the wheel's encoders are connected the Mbed microcontroller via the digital I/O and PWM lines. This board provides high diving current of 9A per channel that is capable of driving Pololu meatal gearhead DC motors of the robot.

Finally, the stepper motor diver of the payload actuation unit is connected to the main microcontroller board via the digital I/O and PWM lines.

2.3.2.1 Payload actuation unit

The payload actuation unit is added to the original design of the two-wheeled robot to provide the payload with the required sliding motion up and down, as will be presented in chapter four. Figure 2.8 shows the structure diagram of the payload actuation unit. It consists of two co-axial parts and a non-captive stepper motor, with its shaft attached to the payload base. The maximum travelling length of the payload actuation unit is around 20 cm as the stepper motor is equipped with a 25cm thread screw shaft as shown in Figure 2.9. The bipolar nema17 was used and according to its technical specifications, listed in appendix A, the motor is able to push a payload with a maximum weight of 4 kg at maximum travelling speed of 10 mm per second.

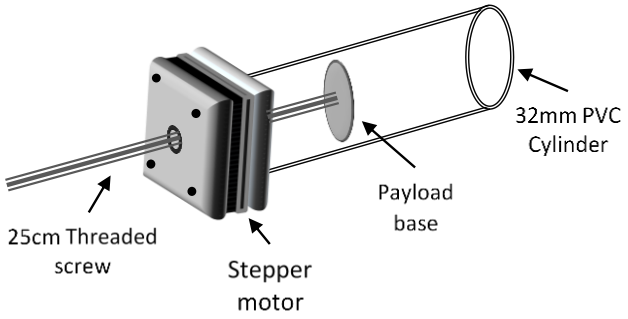


Figure 2.8: ***Payload actuation unit structure***

Figure 2.9: ***Non-captive stepper motor with a thread screw shaft***

2.3.2.2 Effect of the moving payload

Having a dynamically movable payload in the two-wheeled robot system will not add additional DOF to the system but will affect the overall dynamics of the system. The schematic diagram for the new structure of the robot is shown in Figure 2.10. Based on this diagram the dynamic equation of motion of the robot can be expressed as:

$$F = (M_w + m)R^2\ddot{\delta} - mRl\ddot{\theta}\cos(\theta) + mRl\dot{\theta}^2\sin(\theta) + J_w\ddot{\delta} \quad (2.35)$$

$$F_d = ml^2\ddot{\theta} + mlg\sin(\theta) + mRl\ddot{\delta}\cos(\theta) + J_I\ddot{\theta} \quad (2.36)$$

where,

M_w is the mass of the wheels,

m is the mass of the chassis,

R is the radius of the wheel,

δ is the angular position of the wheel,

θ is the tilt angle,

J_w is the moment of inertia of the wheel,

l is the position of the COM,

g is the gravitational acceleration, and

J_I is the moment of inertia of the chassis.

Moving the payload up and down will affect both the global location of COM l_t and the moment of inertia of the chassis J_I. Thus, the global COM of the robot l_t can be calculated as:

$$l_t = \frac{(ml + QM)}{m + M} \quad (2.37)$$

where,

Q is the distance of the payload from the original COM (l), and

M is the mass of the payload.

Similarly, the instantaneous moment of inertia of the robot J_i can be calculated as:

$$J_I = \frac{m(2l)^2}{3} + m(l_t - l)^2 + M(Q - l)^2 \tag{2.38}$$

These changes in the dynamics of the system will affect the performance of the developed control algorithm, therefore adjustments to the controller gain parameters are mandatory to achieve the required response.

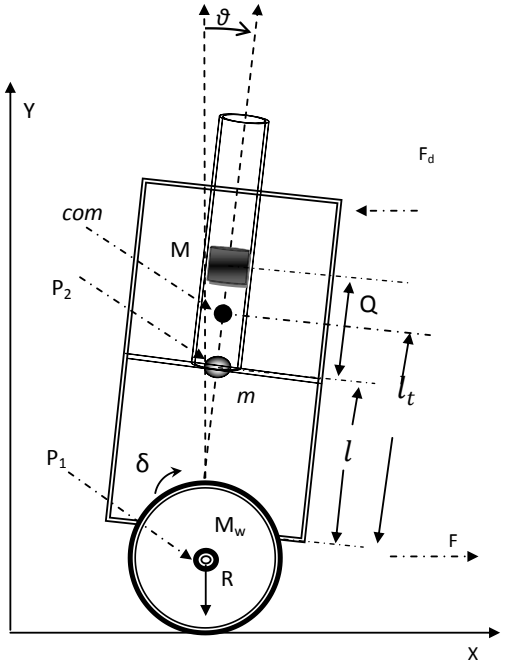


Figure 2.10: ***Two-wheeled robot with moving payload structure***

2.4 Software

A range of software packages and compilers was used in this research in order to develop and realize an appropriate controller for the two-wheeled robot system. This includes simulation and analysis packages such as Matlab and its Simulink environment, Microsoft visual basic, and C/C++ compilers, where the latter have been used in the development and implementation of the core microcontroller code. The following section provides an introduction of the compilers and the developed PC software.

2.4.1 Core microcontroller software

The core microcontroller code was developed and written in C++ language then compiled with a compiler compatible with the target microcontroller board. For instance, the WinAVR was used to develop and implement the control algorithm for the 8-bit AVR development board, whereas the Cygwin cross tool with Linux's gcc-3.3.4 package was employed to compile and implement the code of the two-wheeled robot on the TS-7200 development board. Later, the Mbed platform online compiler was utilized in the development and compilation of the controller code for the Mbed development board, as it will be presented in chapter six.

2.4.2 PC software

PC based software with a graphical user interface (GUI) was specially designed and developed with the Microsoft visual basic to provide an easy interaction between the user and the two-wheeled robot. It connects the computer with the controller of the robot through a wireless Bluetooth serial link.

This software facilitates real-time monitoring of the various parameters of the two-wheeled robot system such as sensors data, position and speed of the robot, gains of the controller, set point and other parameters that can be added. Furthermore, it provides an option to save these data in a csv file for future offline analyses. The PC software was not developed for monitoring and saving data only, but for sending commands to the two-wheeled robot such as sensors calibration, navigation and steering, activation of the payload motion, and controller gain tuning commands. Such functionality is very useful in the controller development and tuning stage as well as during the testing phase of the robot. Figure 2.11 shows a snapshot of the PC software main interface.

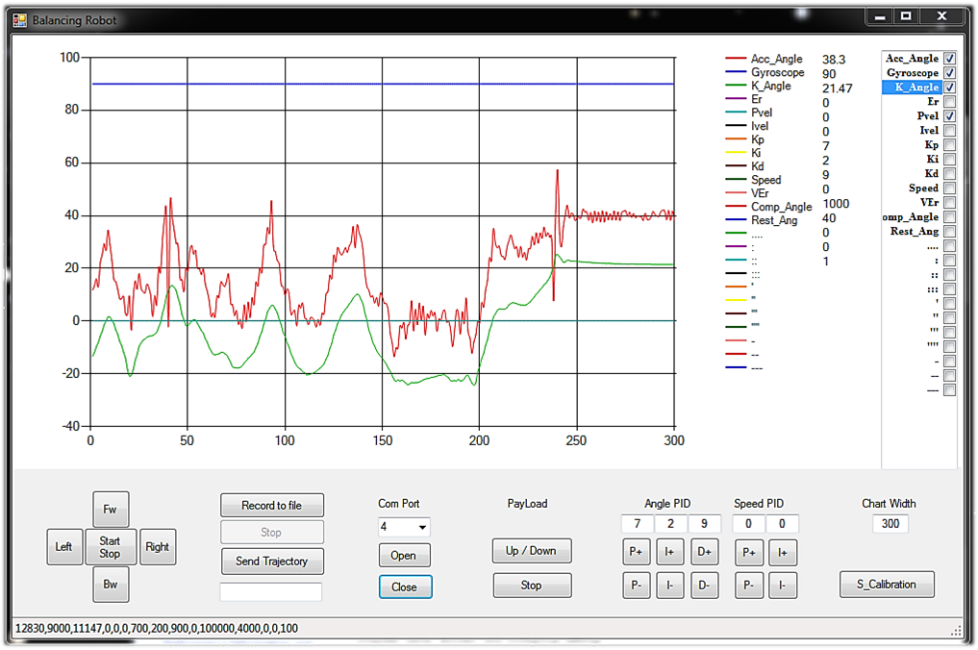


Figure 2.11: ***TWR's PC software GUI snapshot***

2.4.3 Fuzzy data software

Another PC based software was developed in Microsoft visual basic named "Fuzzy data software". This software was mainly developed to help the users and researcher to design the fuzzy controller of the two-wheeled robot. It consists of multiform GUI through which the user can set the number of inputs and outputs, select MSF for every input and output, edit the MSF parameters, and build the fuzzy rules table of the fuzzy controller. Furthermore, the fuzzy data software provides the ability to save the designed fuzzy controller with its rule table in a file on a hard disc and load it later for editing and modification. Figures 2.12 to 2.14 show a snapshot of different windows of the fuzzy data software.

The fuzzy data software generates a binary data file for the fuzzy controller parameters according to specific coding algorithm, explained in chapter five, and transfers it to the microcontroller development board of the robot through the serial port.

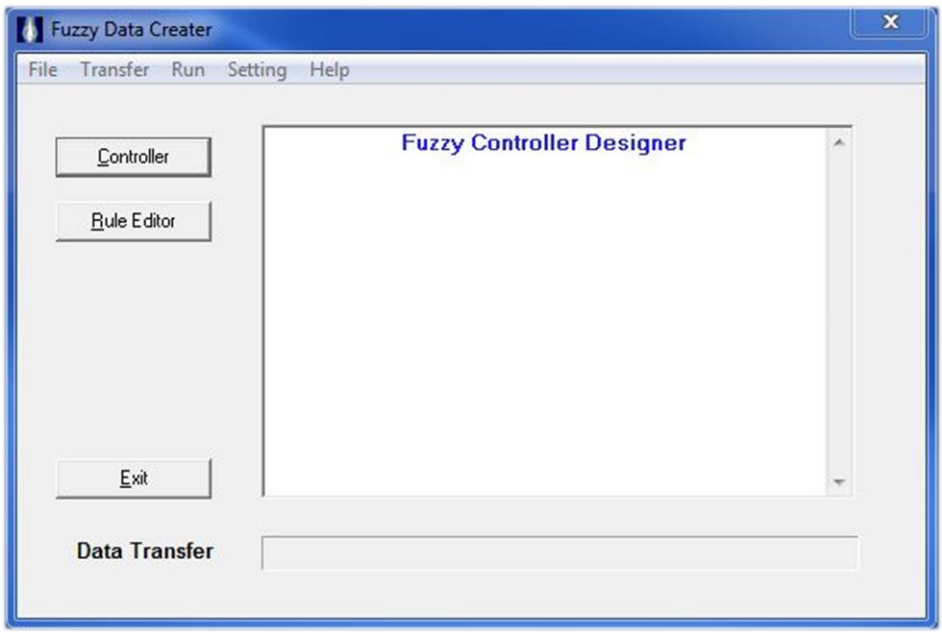


Figure 2.12 ***Fuzzy data PC software main window***

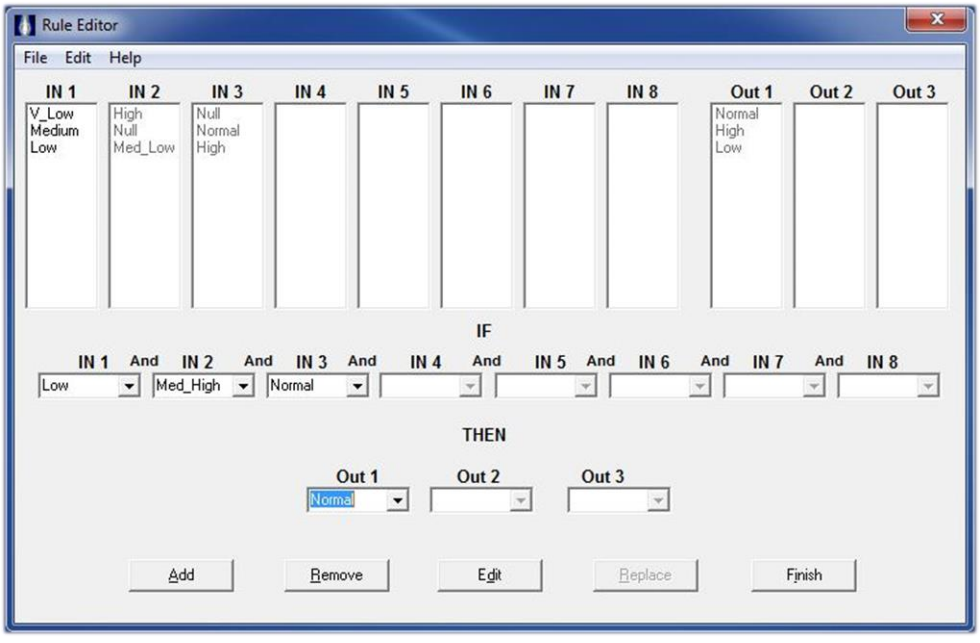
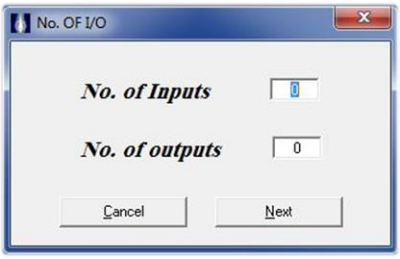
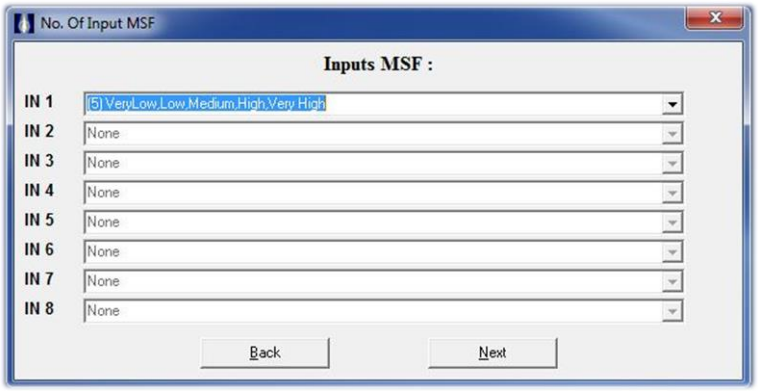


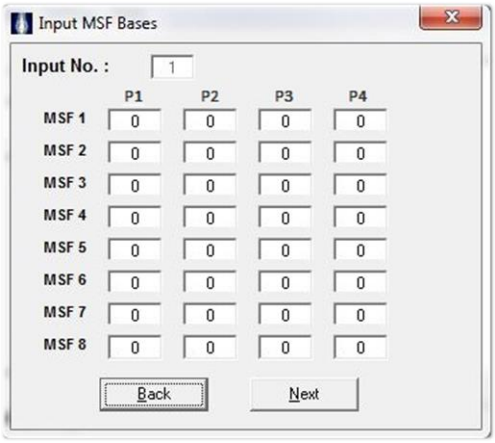
Figure 2.13 ***Fuzzy data PC software Rules editor***



a.



b.



c.

Figure 2.14 ***Fuzzy data PC software controller setting forms***

a. No. input/output selection form, b. Input/output MSF selection form,

c. input/output MSF parameters form

2.5 Summary

In this chapter a description of the two-wheeled robot system has been presented. The hardware and software configurations have also been presented. Briefly, the hardware of the robot consists of Aluminium chassis attached to two driving motors with wheels on both sides. The technical specifications of the electrical parts and electronic circuits of the robot have been presented including the parts have been used later in modifying to improve the robot.

In software, a range of C compliers packages have been addressed. The two-wheeled robot rig communicates with PC via Bluetooth link and transfers the system parameters to the PC software which has been developed as graphical user interface for the system. Special software, Fuzzy data software, has been developed in Microsoft visual basic in order to facilitate the design, coding and implementation of the fuzzy controller on the two-wheeled robot.

CHAPTER 3

Tilt angle measurement of the two wheeled robot

3.1 Introduction

In this chapter, tilt angle measurement techniques of the two-wheeled robot is developed with the assumption that the wheels of the robot is are in contact with the ground. In practice, tilt angle measurement is usually divided into absolute or relative tilt. The absolute tilt is the deviation of an axis or plane measured in relation to the surface or the gravity of the earth. Relative tilt is the deviation with respect to a given reference axis or surface. Absolute tilt is sometimes more difficult to measure than relative tilt especially when there is no physical contact point with the virtual gravity vector or the earth's ground plane. Based on the application and the speed of the measurement, various sensors may be used to measure the absolute tilt angle such as inclinometers, accelerometers, and gyroscopes.

In this work, accurate and fast sensing of the tilt angle of the two wheeled robot is very important to maintain the upright balance. Such accurate measurement could be done by utilizing one type or more of the absolute tilt sensors. Generally, inclinometers have a slow time response which could not follow the dynamics of the system. Hence, accelerometers could be used to measure the tilt angle to provide an accurate steady-state measurement. Alternatively, gyroscopes can also be utilized, as they have a fast response which can follow the fast dynamics of the system.

3.2 Tilt measurement with accelerometers

Accelerometer is a sensor that measures linear acceleration as well as gravity acceleration or the g force. Accelerometers are manufactured in a single or multi-axis configuration with various output types such as analogue, digital, or pulse width modulated output. They are used in many industrial, medical and commercial

applications such as measuring acceleration, vibration monitoring, free fall detection, image stabilization in digital cameras, and tilt sensing application.

3.2.1 Single axis technique

In tilt angle measurement it is preferred to select a low-g with high sensitivity accelerometer in order to achieve the highest resolution of the tilt angle. Tilt is considered as a static measurement while gravity is the acceleration being measured. The relation between the tilt angle and the measured gravity is governed by standard trigonometric functions. Hence, the selection of sensor and the way it will be mounted in the application depends on the full range of the tilt angle to be measured. This will eliminate the nonlinearity of calculation and give the highest possible resolution for that range of tilt.

For example, to obtain the most resolution per degree of change for a range of 0 – 45 degree, the accelerometer sensor should be mounted with the sensing axis which is parallel to the plane of movement as shown in Figure 3.1.

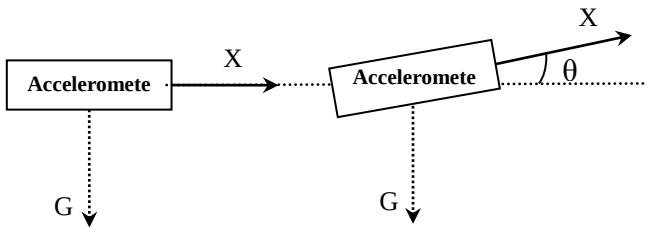


Figure 3.1: ***Single axis accelerometer tilt angle sensing***

Hence, the output of the sensor represents the projection of the gravity vector on the sensing axis, where the amplitude of the measured acceleration changes proportional with the angle between the sensing axis and the ground plane. Thus, the tilt angle from the measured g force, out of the sensor, can be calculated as follows

$$Acc_x = v_{offset} + \left(\frac{\Delta v}{\Delta g} * g * sin(\theta)\right) \quad (3.1)$$

$$\theta = sin^{-1}\left(\frac{Acc_x - v_{offset}}{\frac{\Delta v}{\Delta g}}\right) \quad (3.2)$$

where

Acc_x is the output of the sensor in volts,

v_{offset} is the offset voltage of the output,

$\frac{\Delta v}{\Delta g}$ is the sensitivity of the accelerometer,

g is the earth's gravity acceleration, and

θ is the tilt angle in radians.

It is clearly noticed form the above equations that the sensitivity of the tilt measurements is not constant across the full 360° of tilt angle change due to the non-linearity of the calculation. The sensitivity to the tilt decreases significantly as the tilt angle increases until 90°, when the sensing axis of the accelerometer becomes parallel to the gravity victor as shown in Figure 3.2.

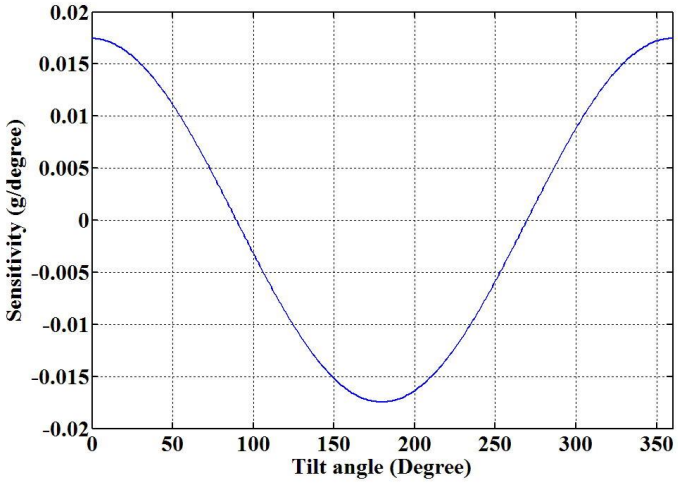


Figure 3.2: ***Tilt angle measurement sensitivity with single-axis accelerometer***

3.2.2 Multi axis technique

In order to improve the tilt sensitivity and enhance the accuracy of the tilt angle measurement in the single axis technique, multi-axis sensing technique is adopted. In this technique a multi-axis accelerometer is used and mounted with the same

considerations of the single-axis technique. Figure 3.3 shows a dual-axis accelerometer tilt sensing configuration.

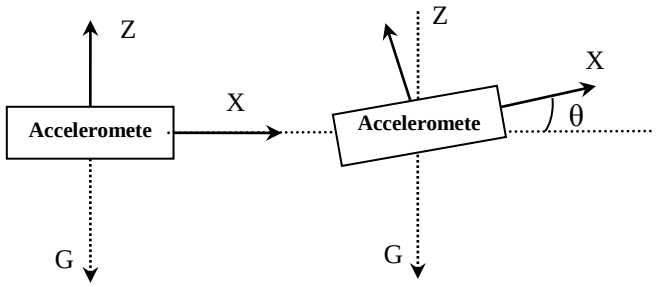


Figure 3.3: ***Dual axis accelerometer tilt angle Sensing technique***

In this configuration, both axes will have change in the sensitivity to the tilt but in opposite direction as X-axis sensitivity decreases Z-axis will increase. Combining the measurement of both axes will provide a constant sensitivity. The tilt angle calculation in this configuration can be expressed as follows

$$Acc_x = v_{offset} + \left(\frac{\Delta v}{\Delta g} * g * sin(\theta)\right) \quad (3.3)$$

$$Acc_z = v_{offset} + \left(\frac{\Delta v}{\Delta g} * g * cos(\theta)\right) \quad (3.4)$$

$$\theta = tan^{-1}\left(\frac{Acc_x}{Acc_z}\right) \quad (3.5)$$

where

Acc_x is the X-axis output of the sensor,

Acc_z is the Z-axis output of the sensor,

v_{offset} is the offset voltage of the output,

$\frac{\Delta v}{\Delta g}$ is the sensitivity of the accelerometer,

g is the earth's gravity acceleration, and

θ is the tilt angle in radians.

Based on the same configuration of multi-axis technique, triple axis accelerometers can also be utilized to measure the tilt angle with constant sensitivity. In this configuration both of the tilt (pitch) angle and the roll angle can be measured with one sensor as

$$Pitch\ \theta = tan^{-1}\left(\frac{Acc_x}{\sqrt{\left(Acc_y\right)^2 + (Acc_z)^2}}\right) \quad (3.6)$$

$$Roll\ \alpha = tan^{-1}\left(\frac{Acc_y}{\sqrt{\left(Acc_x\ \right)^2 + (Acc_z)^2}}\right) \quad (3.7)$$

where

Acc_x is the calibrated X-axis acceleration,

Acc_y is the calibrated Y-axis acceleration,

Acc_z is the calibrated Z-axis acceleration,

θ is the pitch (tilt) angle in radians, and

α is the roll angle in radians.

Figure 3.4 shows the tilt measurement sensitivity of the multi-axis technique vs the single-axis.

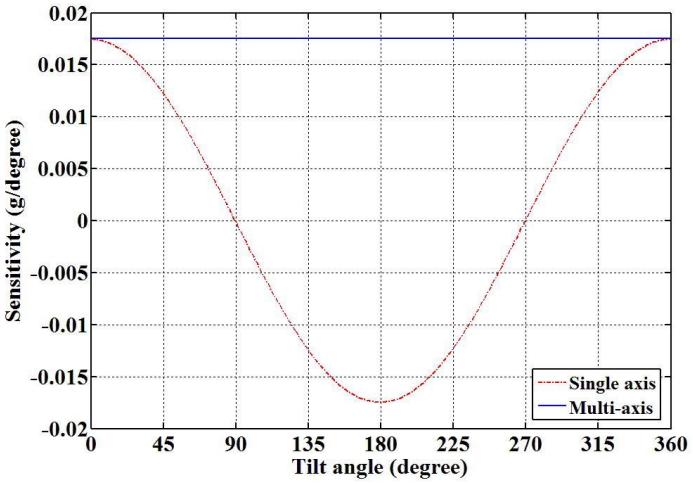
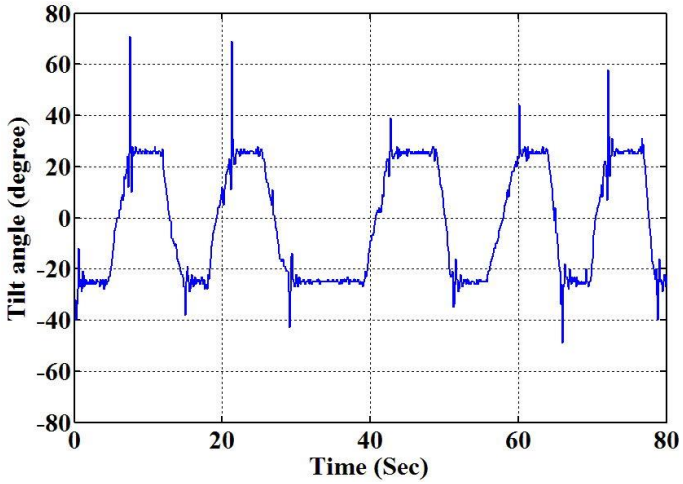


Figure 3.4: ***Tilt angle measurement sensitivity***

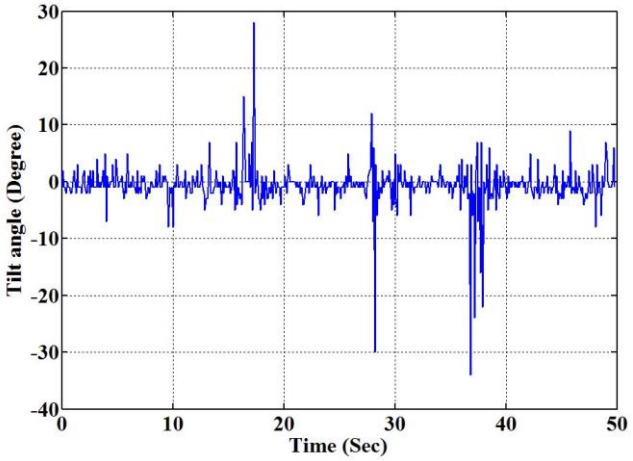
3.2.3 Implementation

In this work, the ADXL203 accelerometer was used for tilt angle measurement of the two-wheeled robot. It has two axes with sensitivity of ±1.7g. It produces analogue voltage outputs that are proportional to the acceleration of each axis. Each

of these outputs is connected to one of the 10-bit ADC channels of the microcontroller then digitally calibrated by taking the offset voltage at zero acceleration from the final reading. The accelerometer is positioned at the centre of the middle layer of the robot's chassis with one of its axes aligned with z-axis of the robot, while the other axis of the accelerometer is aligned with x-axis, the forward / backward movement direction, of the robot. Such orientation facilitates the implementation of both single axis and multi-axis tilt angle measurement techniques. Figure 3.5 shows the experimental results of tilt angle measurement with accelerometer using two-axis technique with sampling rate of 100 Hz.



a



b

Figure 3.5: ***Tilt angle measurements with 2-axis accelerometer sensor a) ±20°, b) ±2°***

It is clearly noticed that the measurements are corrupted by noise. This is because the tilt calculation with accelerometers was based on the measured acceleration, gravity force in steady state. However the measured acceleration was influenced by the movements of the two-wheeled robot, so the accelerometer tends to sense the horizontal acceleration as well.

3.3 Tilt measurement with gyroscopes

Gyroscope is a mechanical device for measuring or maintaining orientation which is based on the principle of preserving angular momentum. Gyroscopes typically comprise a spinning wheel or disc, in which the axle is free, to assume any orientation. However, after the development of the MEMS technology, Gyroscopes are manufactured in very small sizes and light weight that can fit in a small chip with options of multi-axis measurement configuration.

Gyroscope can be utilized for tilt angle sensing, as it facilitates orientation calculation and measuring the rate of change of angle around the rotation axis. Hence, the tilt angle can be calculated by integrating the calibrated gyroscope reading.

3.3.1 Implementation

In this work the ADXRS614 gyroscope sensor was utilized for tilt angle measurement of the two-wheeled robot. It is a single axis sensor of ±30°/sec measurement range with an analogue voltage output. The output is connected to the ADC channel of the microcontroller then digitally calibrated by taking the zero offset voltage of the reading. The gyroscope is mounted at the centre of the middle layer of the robot's chassis with its sensitive axis parallel to wheels axis of the robot. Hence, the tilt angle of the two-wheeled robot can be obtained by integrating the gyroscope reading. The experimental result of tilt angle measurement with the gyroscope, shown in Figure 3.6, reveals that there is a drift over time in the tilt measurements due to an integrational error.

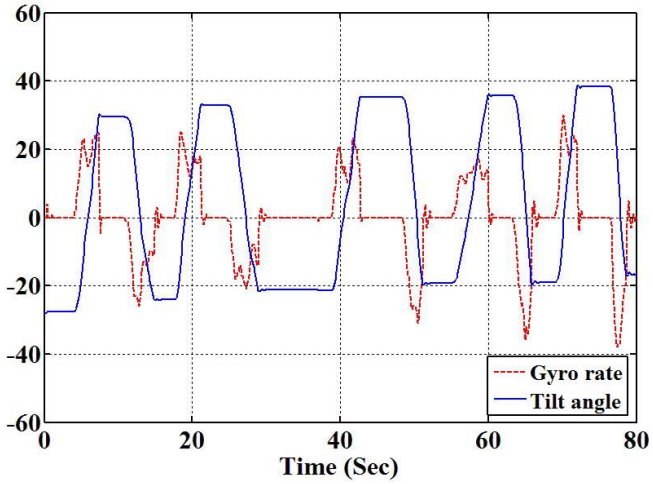


Figure 3.6: ***Tilt angle measurements with gyroscope sensor***

3.4 Sensor fusion

Sensor or data fusion is a technique for delivering a robust and comprehensive information of an application or process of interest by fusing data from various different sources. Sensor fusion is significant to any application that require decisions to be made on a high quality information obtained from large amounts of data. Data fusion techniques are adopted in many applications in industry, military and science such as radar systems, civilian surveillance and monitoring, and in industrial process control. In principle, automated data fusion processes allow essential measurements and information to be combined to provide knowledge of sufficient richness and further for decisions to be formulated and executed autonomously (Higgins 1975).

Accurate and fast measurement of the tilt angle is of special significance in the two-wheeled robot system to maintain upright balance of the robot; such accurate measurement could be done using expensive IMU. However, replacing this unit with a low cost MEMS sensor cause significant error in the measurement. For instance, inclinometers have slow response time, and could not follow the robot dynamics. As discussed in the previous section, an accelerometer provides accurate steady-state measurement, but the measurement will be noisy due to movements of the robot. On the other hand, a gyroscope has a fast response that can follow the dynamics of the two-wheeled robot, but suffers from time drift. Thus, a sensor fusion technique is considered to obtain accurate angle information (Hyung-Jik and

Seul 2009) (Lee and Jung 2012). The following sections introduce the concept of multi-sensor fusion.

3.4.1 Complementary filter

The complementary filter is one of the data fusion techniques. It comprises of two simple filters; low pass and high pass. Hence it is often implemented to combine two sources of data affected by noise at different bandwidth to obtain clear information.

The tilt angle measurements using accelerometer is affected by high frequency oscillation due to the movements of the robot, and the tilt angle data from the gyroscope is affected by integration drift, which is low frequency. In this case, the complementary filter will be an appropriate sensor fusion technique to combine measured data from both sensors and obtain accurate tilt angle information.

In this technique, the low pass filter part acts on the tilt data from the accelerometer to eliminate the high frequency components considered as noise. Similarly, the high pass filter part acts on the tilt data from the gyroscope eliminating the low frequency components which represent the integral drift noise. Then, the outputs of the low pass and high pass filters are added together to form tilt information which is free of oscillation noise and drift. Figure 3.7 illustrates the structure of the complementary filter. The mathematical representation of the complimentary filter is described through the following set of equations.

$$\theta = \theta\, G(s) + w\, \frac{\left(1 - G(s)\right)}{s} \tag{3.8}$$

$$G(s) = \frac{1}{\tau s + 1} \tag{3.9}$$

$$1 - G(s) = \frac{\tau s}{\tau s + 1} \tag{3.10}$$

where

G(s) is the low pass filter transfer function,

1-G(s) is the high pass filter transfer function,

θ	is the tilt data from the accelerometer, and
w	is the gyroscope sensor data.

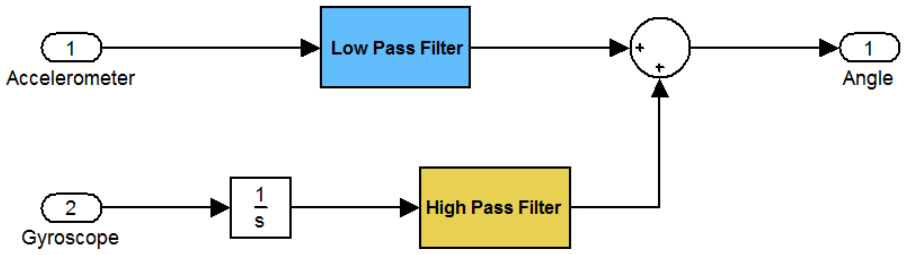


Figure 3.7: ***Structure of the complementary filter***

The cut-off frequency considered to be the same for both of the low pass filter and the high pass filter. This frequency acts like a threshold between trusting the gyroscope measurement and trusting the accelerometer measurement. Therefore, the time constant or the cut-off frequency of the filter is a tuneable parameter. It mainly depends on the performance of the sensors in terms of accelerometer noise and gyroscope drift.

3.4.1.1 Implementation

The complementary filter can be easily implemented in real-time digital systems, as a first order discrete-time filter as

$$Y(k) = a\,Y(k-1) + X(k) \tag{3.11}$$

where

Y(k) is the output of the filter,
X(k) is the input of the filter,
a is the filter coefficient, and
dt is the sampling time.

Hence, the full structure of the complimentary filter can be expressed as

$$Y(k) = a\left(Y(k-1) + X_{Gy}(k) * dt\right) + (1-a)X_{Acc}(k) \tag{3.12}$$

$$a = \frac{\tau}{\tau + dt} \tag{3.13}$$

where

Y(k) is the tilt output of the filter,

$X_{Gy}(k)$ is the gyroscope data,

$X_{Acc}(k)$ is the accelerometer data,

a is the filter coefficient,

τ is the time constant, and

dt is the sampling time.

In the above expression the first part of the equation represents the high pass filter part with digital integration while the second part represents the low pass filter part.

Accordingly, the important factor of the filter performance is the time constant τ of the filter, the corresponding proper cut-off frequency. This time constant can be found by investigating the characteristics of the sensors experimentally. For instance, if the integral drift in the gyroscope angle measurement is about 2° per second, then the time constant could be any value smaller than one second, so the filter output would never drifted more than two degrees. Generally, the lower the time constant the lower the drift in the output; however, decreasing the time constant will increase the output noise from the accelerometer measurement. Therefore, selecting the time constant is subject to the application, and there is always a trade-off between the accelerometer noise and the gyroscope drift.

Based on the sensors specification and the tilt sensing experimental results in this work, the appropriate time constant of the complimentary filter was calculated as τ = 0.5 Sec for sampling rate of 100 samples per second. Figures 3.8 and 3.9 show the tilt angle measurement with the complementary filter. As noted, it is relatively free of measurement noise and integration drift.

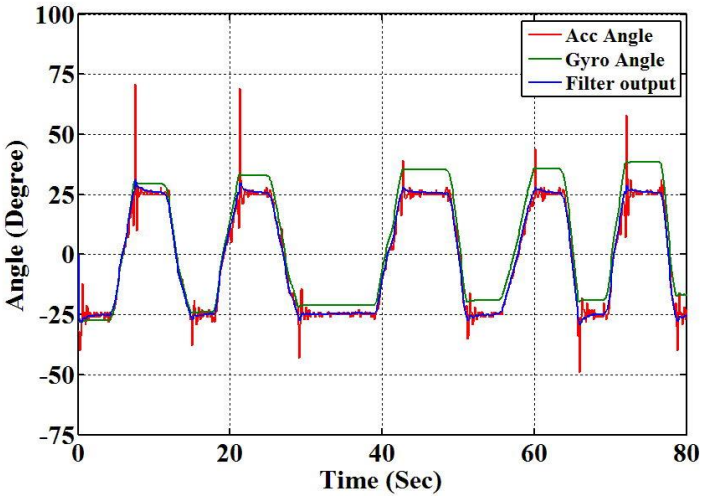


Figure 3.8: ***Tilt angle measurement: complementary filter vs accelerometer and gyroscope at ±25°***

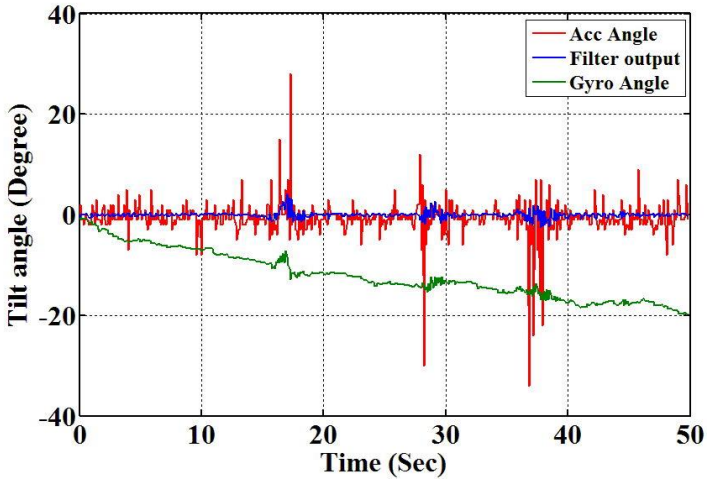


Figure 3.9: ***Tilt angle measurement: complementary filter vs accelerometer and gyroscope at ±2°***

3.4.2 Kalman filter

Kalman filter was presented by Rudolf Kalman (1960) as a solution to the discrete-data linear filtering problem. It is a recursive algorithm for variables estimation based on a chain of noisy and inaccurate measurements taken over time. These estimates generally would be more accurate than taking an individual measurement. Since then Kalman filter has been applied to many applications such as target

tracking, robot localization, and map building from range sensors due to its good practical results and its suitability for online real-time processing. Kalman filter has been used for sensor fusion, and it is one of the main elements in motion planning and control in robotic applications.

In this research, the Kalman filter was considered as an alternative sensor fusion technique to achieve accurate tilt angle measurement. Then Kalman filter takes the noisy measurements from the accelerometer and the gyroscope, state of the system, in order to estimate the tilt angle of the two-wheeled robot based on the current and previous states. The state equation of the system can be written as:

$$x_{k+1} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x_k + w_k \tag{3.14}$$

$$z_k = \begin{bmatrix} 1 & 0 \end{bmatrix} x_k + v_k \tag{3.15}$$

where,

$x_k = \begin{bmatrix} \theta \\ \dot{\theta}_b \end{bmatrix}$ is the states of the system,

w_k is zero mean white process noise,

z_k is the observation or measurement, and

v_k is zero mean white measurement noise.

The Kalman filtering process involves two sets of calculation one for prediction and the other for updating and correcting the measurements. In the prediction phase, the current state is estimated based on the gyro measurement and all the previous states. Then the a *priori* error covariance matrix P is estimated based on the old error covariance matrix as

$$\widehat{x}_{k|k-1} = F\widehat{x}_{k|-1k-1} + B\dot{\theta}_k \tag{3.16}$$

$$P_{k|k-1} = FP_{k-1|k-1}F^T + Q_k \tag{3.17}$$

where, Q_k is the covariance of the process noise.

The matrix P is used to estimate how much the current values of the estimated state are trusted. The current estimated state is more trusted for the smaller value of P. The error covariance matrix P in this study is a 2×2 matrix.

In the second phase of the Kalman filter calculation, the error between the actual measurement and the estimated values is calculated as

$$\tilde{y}_k = z_k - H\hat{x}_{k|k-1} \tag{3.18}$$

Then the innovation covariance *S* is calculated as

$$S_k = HP_{k|k-1}H^T + R \tag{3.19}$$

where, R_k is the covariance of the measurement noise.

The larger the value of *S* means the bigger the value of the measurement noise and the incoming measurement is not trusted that much.

At this point the Kalman gain matrix can be calculated based on the *P* matrix and *S* value as

$$K_k = P_{k|k-1}H^T S_k^{-1} \tag{3.20}$$

The last step in the update phase is updating the posteriori error covariance matrix *P* using the calculated Kalman gain matrix as follows.

$$P_{k|k|} = (I - K_k H)P_{k|k-1} \tag{3.21}$$

3.4.2.1 Implementation

The two phases of the Kalman filter process, predict and update, were coded in C++ and implemented on the Mbed microcontroller to obtain the tilt angle of the two-wheeled robot. Figure 3.10 illustrates the programming code for the implementation of Kalman filter.

The filter was tuned by setting the process noise Q_K and the measurement noise R_K values. Choosing the right value is essential to ensure the best performance of the Kalman filter. For instance, the estimated angle will drift for insufficient value of $Q_{\dot{\theta}}$, and for large value of Q_θ the response of the filter will be slowed as the estimated angle will be trusted too much. Also the filter would be less responsive for a high value of the measurement noise *R* as the new measurements are not being used too much. Moreover the output may have an overshoot and become noisy for a very small *R* value due to the use of the accelerometer measurement.

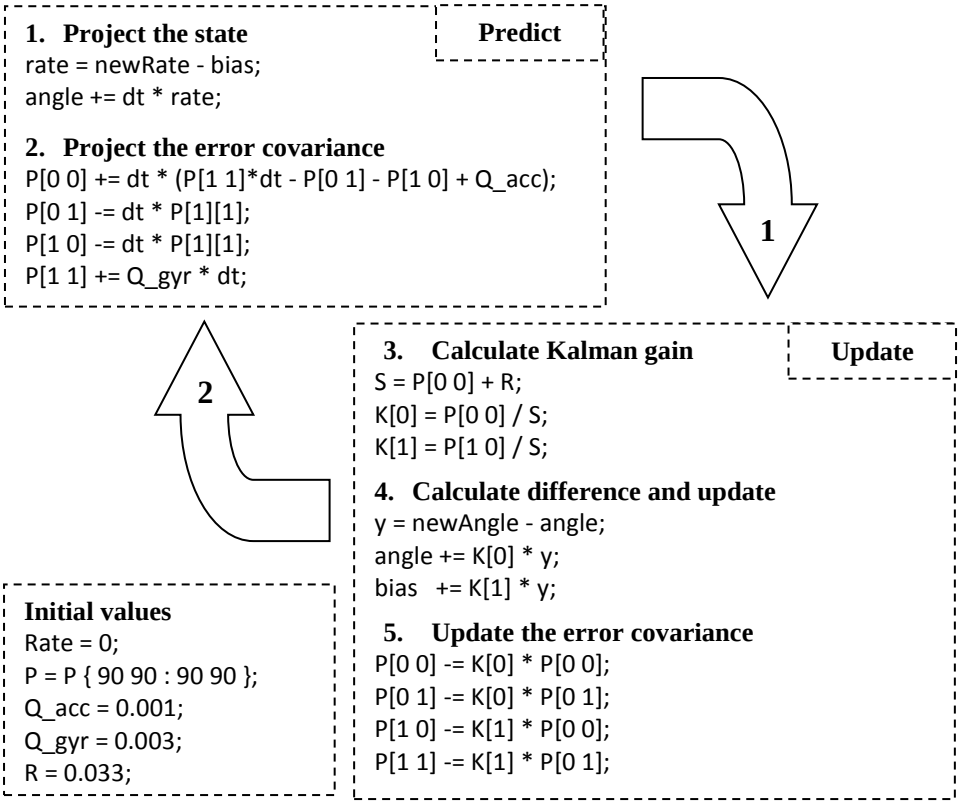


Figure 3.10: ***The operation diagram of the Kalman filter***

Figures 3.11 and 3.12 show the Kalman filter tilt angle measurement. It is clearly noted in Figure 3.11 that there was an overshoot in the tilt measurements due to the improper values of Q_K and R. Whereas Figure 3.12 shows the tilt measurement of a tuned Kalman filter compared with both angle measurements from the gyroscope data and the accelerometer. The angle from the gyroscope suffered sufficient drift, and the angle from the accelerometer was corrupted by noise and sharp spikes.

Another comparison was made between the tilt angle measurement of the complementary filter and the Kalman filter during balancing the two-wheeled robot as in Figure 3.13. It is noted that Kalman filter reduced the noise in the tilt measurement and successfully eliminated the small drift in the complementary filter output.

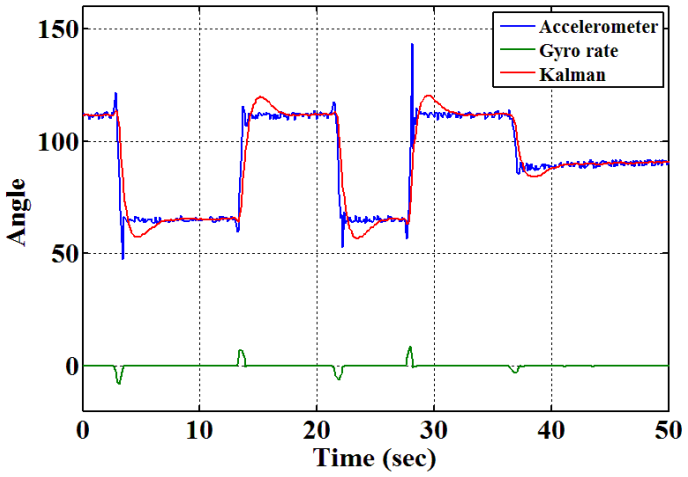


Figure 3.11: ***Tilt measurements of an un-tuned Kalman filter***

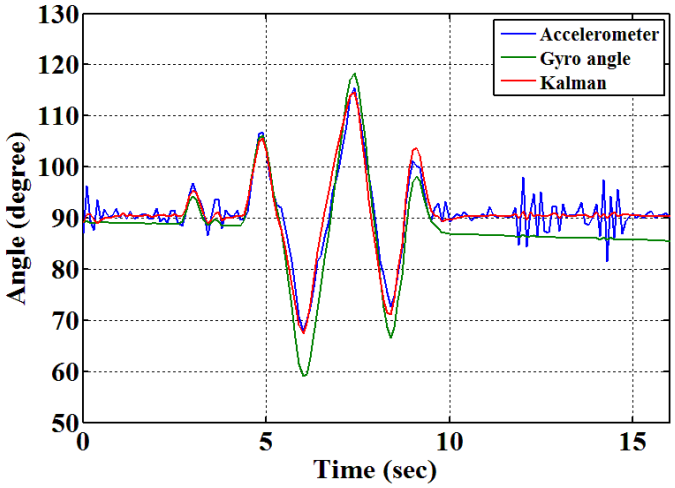


Figure 3.12: ***Tilt measurements of a tuned Kalman filter***

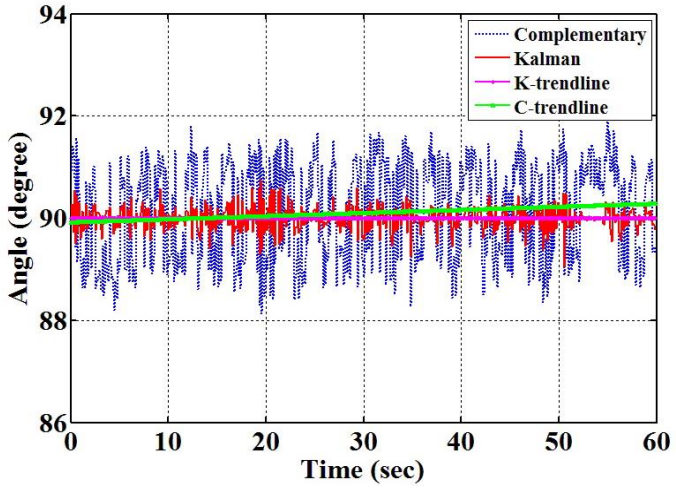


Figure 3.13: ***Tilt angle measurement, Kalman Vs complementary filter***

3.5 Position and speed measurement

The position and travel speed of the two-wheeled robot are another state to be controlled. It is mainly linked with the wheel size and the speed of each wheel of the robot. Hence, an encoder is utilized to measure the speed of each wheel, which generates 360 pulses per one full wheel revolution. The output of both wheels encoders are connected to motor driver board where the pulses are converted to speed in revolution per minute RPM units. The microcontroller reads the speed of the wheels from the motor driver board over I^2C bus then calculates the linear (travel) speed and the relative position of the robot. This is based on the diameter and the speed of the wheels as shown below, assuming that both wheels rotate at the same speed.

$$S = \frac{2\pi R_w N}{60} \tag{3.22}$$

$$D = \frac{2\pi R_w N}{60}\Delta t \tag{3.23}$$

where

S is the travel speed of the robot in (m/sec),

R_w is the radius of the wheel,

N is the average speed of the left and right wheel in RPM units,

D is the travelled distance of the robot, and

Δt is the sampling time.

3.6 Summary

In this chapter the techniques of tilt angle measurement have been presented. A description of the tilt sensors with the associated problems such as offset, noise and drift have been described. Accurate measurements of the tilt angle have been obtained, by utilizing sensor fusion mechanisms. Complimentary filter has been realised for tilt angle sensing based on filtering the accelerometer and gyroscope measurements. Another sensor fusion technique has been presented and realized with Kalman filter, in order to enhance the tilt angle measurement which is important to achieve a better balance performance.

In the subsequent chapters the real-time implementation of the control algorithms for the two-wheeled robot will be discussed.

CHAPTER 4

Controller design and implementation of the two wheeled robot

4.1 Introduction

This chapter is devoted to the design and real time-implementation of a controller for the two-wheeled robot. First, the design aspects of the two-wheeled robot controller based on the PID controller are presented. Followed by the digital implementation, tuning, and assessment of the controller for upright balance and position control. Then, steering control of the two-wheeled robot is achieved by adopting a differential steering mechanism. Also, the PID controller of the two-wheeled robot is modified in order to provide the ability of self-standing from the rest position followed by the real-time implementation on the robot hardware and performance evaluation

Secondly, a dynamic payload position is considered as well by adding a linear actuator in order to activate the payload to move along the intermediate body with appropriate speeds as described in chapter 2. Consequently, the addition of the new actuator to the two-wheeled robot system increases the DOF along the intermediate body of the system. Therefore, further development of the control algorithm of the system is also required. Hence, a new controller structure for balancing and positioning the two-wheel robot is developed and presented in this chapter.

Finally, the new controller structure of the two-wheel robot is tested for balancing the robot on an inclined surfaces where three inclination angles are considered.

4.2 Close loop response

In this section, the closed loop response of the two-wheeled robot system is tested with a PID controller. Although the mathematical model showed that the two-wheeled robot system is non-linear, PID or other linear controller can be used within the linear region of operation. The main objective of the PID controller is to eliminate or minimize the error between the desired and the actual output of the system. Generally, PID controllers are very simple to implement and provide adequate results, therefore they are the most commonly used controller in industry (Ogata, 2002).

The basic PID controller consists of three parts with tuneable gain parameters (K_p, K_i and K_d) that are associated with the proportional error, integral of the error and the change of error respectively, refer to Figure 4.1. The proportional part regulates the control signal to reduce the error and the rise time of the output to reach the desired value. The second part, the integral, adjusts the control signal based on the accumulative error and tends to eliminate the steady state error of the response. While the derivative part uses the change of error in the system in order to reduce the overshoot in the response. Combining all three terms weighted by their gains to generate the control signal of the system, the system will benefit from the individual advantages.

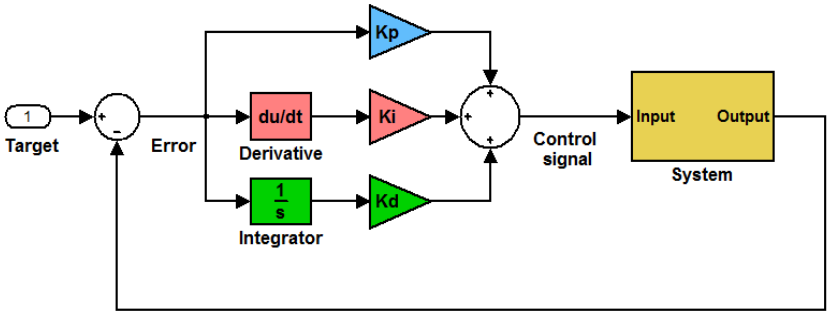


Figure 4.1 ***PID controller structure diagram***

Practically, the presence of the three controller parts depends on the application and the required system response, therefore the controller may be called by the employed parts such as PI, PD, P, or I controller to reflect the absence of the other parts. PI controllers are more common in the industry for two reasons: first,

having the integral part will ensure that the system will reach its target and there is no steady state error. Second, the derivative part of the controller has high sensitivity to measurement noise and may produce oscillation in the output of the system that may cause mechanical damage (Ogata, 2002).

4.3 Controller design of the system

In order to design a full controller for the two-wheeled robot system model, a couple of assumptions are made. Firstly, the parameters of the mathematical model were chosen to be the same as the actual robot prototype as listed in Table 4.1. Secondly, there is a full contact and no slipping between the wheels and the ground all the time. Finally, both of the model inputs are driven by the same control signal, therefore the robot will only move forward and backward. Based on these assumptions two PID based control loops are designed. The first loop, PD controller, is responsible for maintaining balance of the robot by keeping the tilt angle of the chassis close to zero with z-axis. While the second loop, PD controller, is controlling the horizontal movement and the position of the robot. Figure 4.2 depicts the full diagram of the designed control system.

Table 4.1: ***Robot's model parameters***

Parameter	Symbol	Value
Mass of the motors and wheels	M_w	3 Kg
Mass of the chassis	m_I	1.4 Kg
Wheel radius	R	0.05 m
Centre of gravity position	l	0.15 m
Inertia of the chassis	J_I	0.0225
Inertia of the wheel	J_w	0.0002
Inertia of the motor	J_m	0.002
Resistance of the motor	R_a	3 Ω
Velocity constant of the motor	K_v	0.3
Torque constant of the motor	K_m	0.3
Gear ratio	N	1 : 29

The gains of the two PD controllers have to be tuned simultaneously due to the strong coupling between the two control loops. At this point, the gain parameters were manually tuned using a heuristic approach to K_P =20, K_D =28 for the position control loop and K_P =114, K_D =61 for the angle control loop. The

controller successfully stabilized the robot with a satisfactory results, Figurers 4.3 to 4.5, in terms of tilt angle and position of the robot.

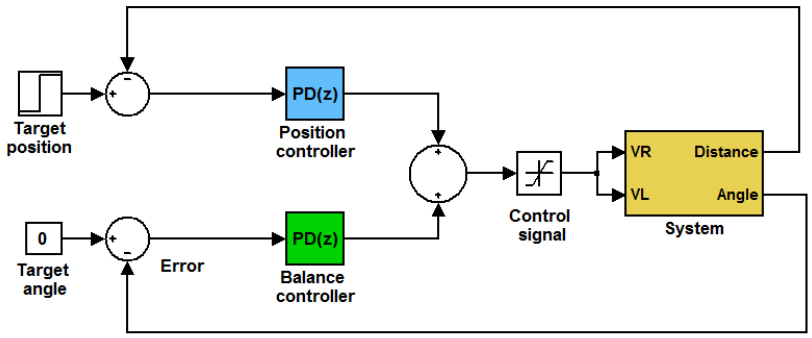


Figure 4.2 ***Close loop control diagram of the two-wheeled robot system***

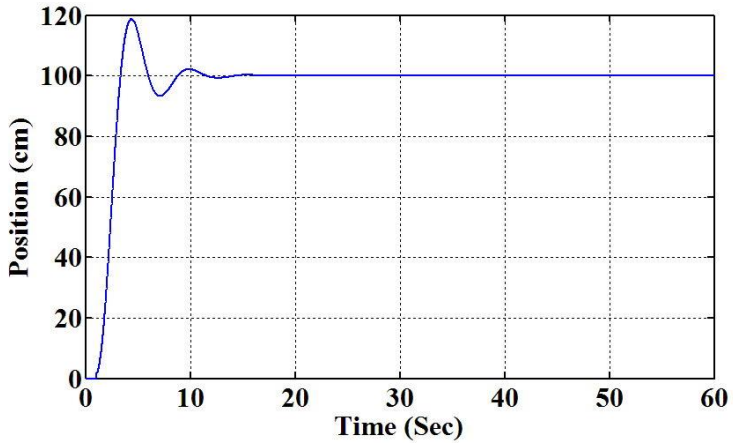


Figure 4.3 ***Travelled distance of the robot in closed loop configuration***

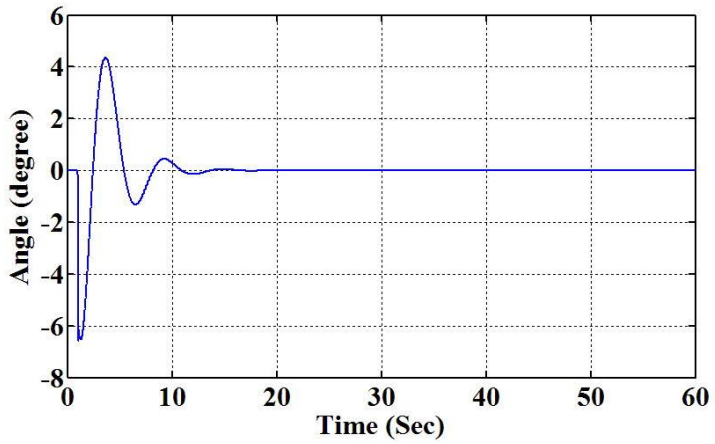


Figure 4.4: ***Tilt angle of the robot in closed loop configuration***

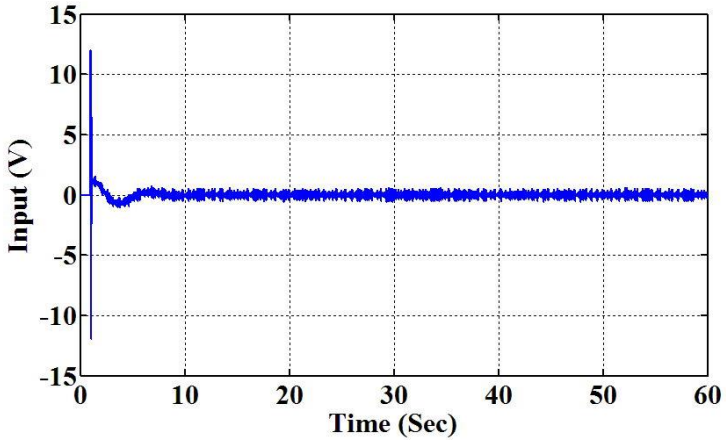


Figure 4.5: ***Left and right wheel motor input of the robot in closed loop configuration***

4.3.1 Disturbances test

External disturbances were simulated in order to test the robustness of the designed controller. Disturbances represented by two consecutive pulses opposite to each other with same duration and deferent amplitude. The first pulse was applied to the system after 20 seconds for a duration of one second, while the second pulse with 50% of the first pulse amplitude was applied after 40 seconds. In order to test each of the control loops of the system, the disturbance signal was added only to the feedback of the balancing controller. The simulation results with disturbance signal profile, depicted in Figures 4.6 and 4.7, show that the controller was able to maintain the balance of the robot in the presence of the disturbances, as the tilt angle successfully returned to zero after each pulse. Although the disturbance signal was not applied to the position controller, there was a change in the position of the robot associated with each pulse due to natural coupling between the variables of the two-wheeled robot system.

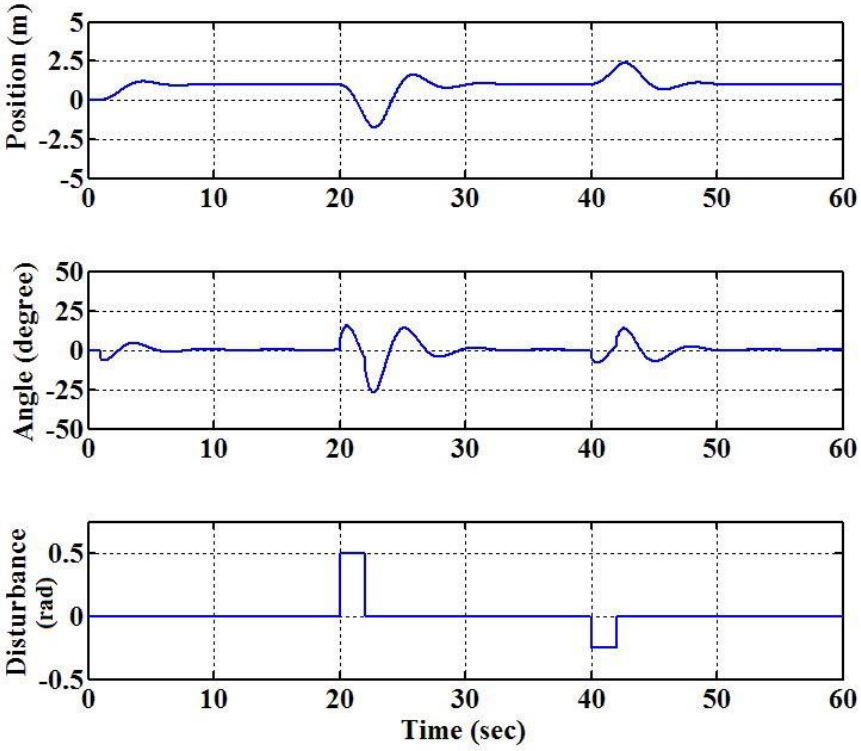


Figure 4.6: ***Closed loop response with disturbance applied to the balancing control loop***

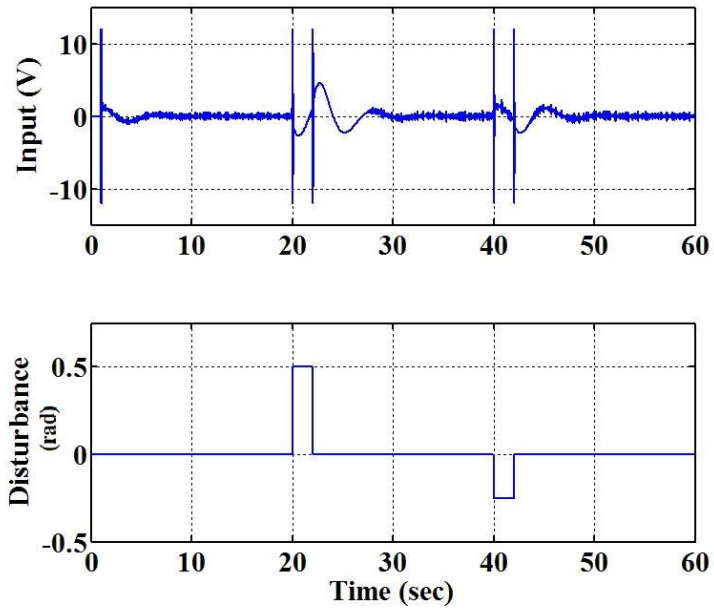


Figure 4.7: ***Motors input with disturbance applied to the balancing control loop***

Similarly, a disturbance simulation test was conducted for the second control loop, the position controller, using the same disturbance signal profile which was added to the feedback signal. The results revealed that the controller was able to keep the robot at the target position after every disturbance pulse as shown in Figures 4.8 and 4.9.

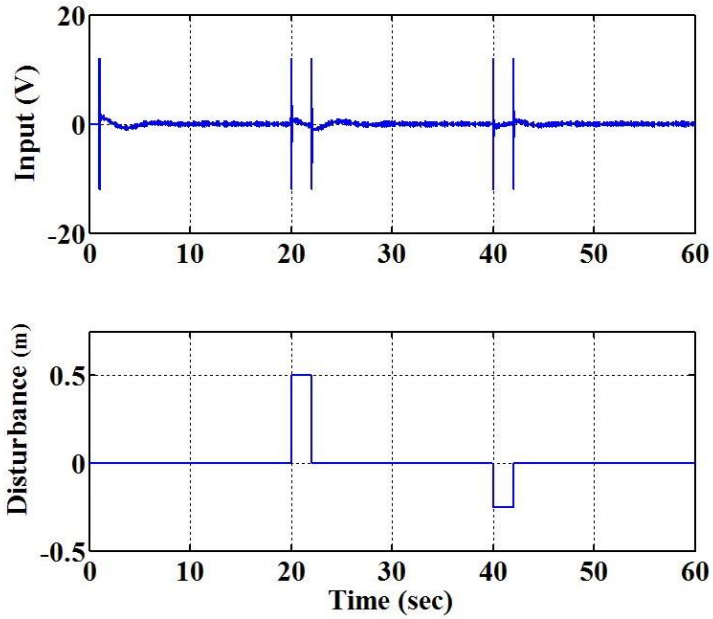


Figure 4.8: ***Motors input with disturbance applied to the position control loop***

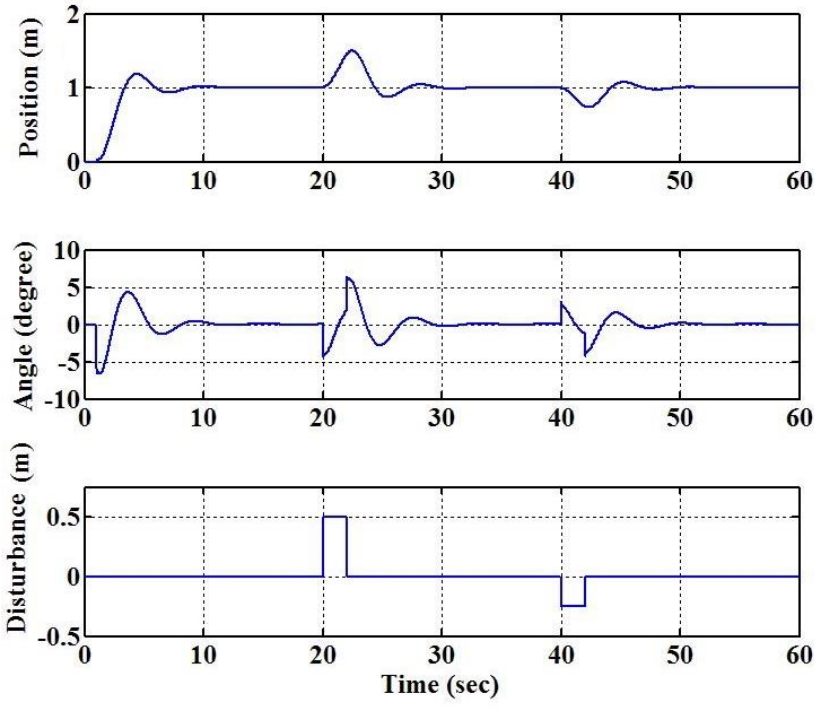


Figure 4.9: ***Closed loop response with disturbance applied to the position control loop***

4.3.2 The effect of wheel size

One of the key design parameters of the two-wheeled robot is the size and the radius of the wheels. In order to understand the influence of the wheel size on the controller of the two-wheeled robot system, the system was simulated with three deferent wheel sizes with radii 2.5cm, 5cm, and 7.5cm.

Comparing the simulation results in Figure 4.10 reveal that the response of the tilt angle is slightly slower with a smaller overshoot for bigger wheel size. However, having a bigger wheel size will increase the error and the overshoot was increased by about 10% in the position response of the two-wheeled robot as shown in Figure 4.11.

On the other hand, the results of the input of the motors and the control signal in Figure 4.12 show that the oscillation in the signal increased for a small wheel size. Such results imply that the two-wheeled robot will require less energy about 50% less to maintain balance for a bigger wheel size as listed in table 4.2. Thus the wheel size should be chosen to compromise between the required response and the energy consumption.

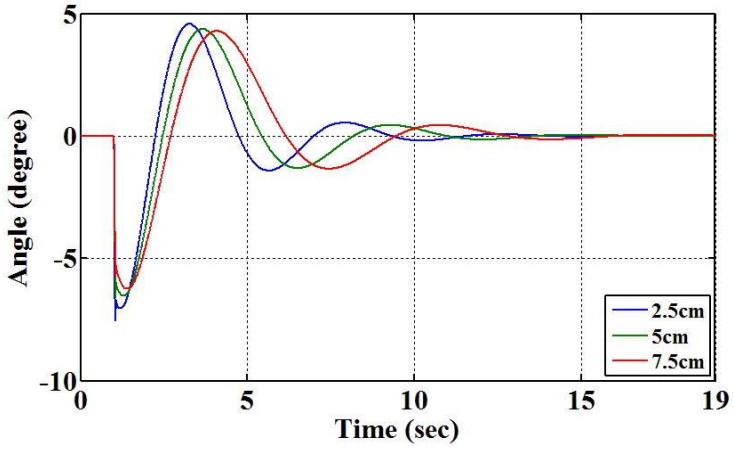


Figure 4.10: ***Tilt angle for different wheel size***

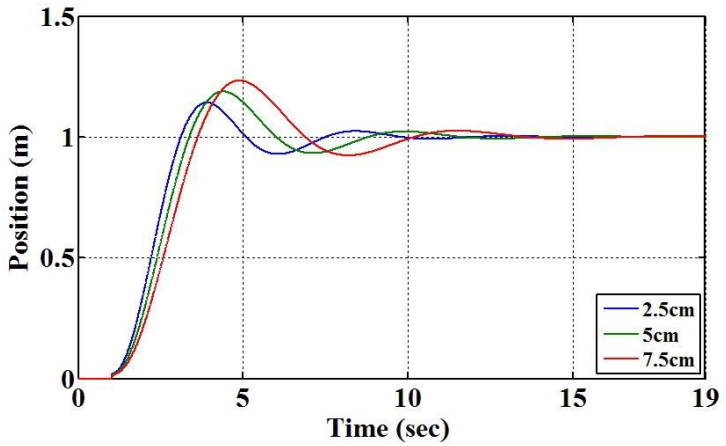


Figure 4.11: ***Travelled distance for different wheel size***

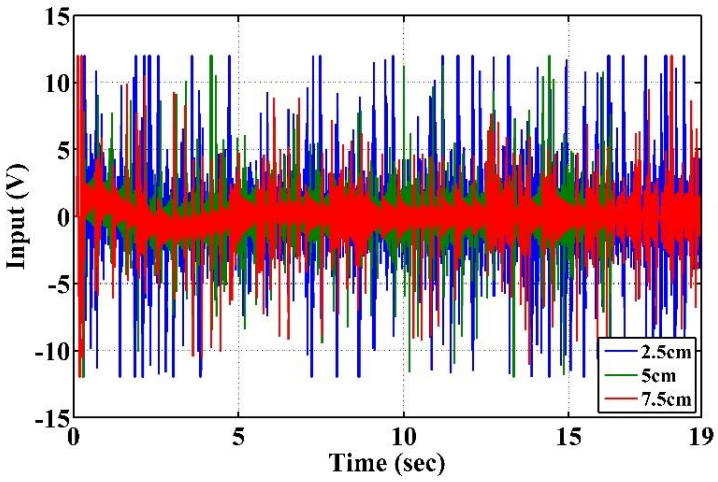


Figure 4.12: ***Motors input for different wheel size***

Table 4.2: ***Two wheels robot response analysis with different wheels size***

Wheel size		Rise time	Settling time	Over shoot	MAE	RSME	Equivalent energy
25 cm	Position	2.38	8.74	14.28%	0.13126	0.28533	-
	Angle	-	4.81	8.46%	0.02320	0.04013	-
	Input	-	-	-	3.34431	4.69886	334.2153
50 cm	Position	1.82	7.95	18.69%	0.11599	0.26857	-
	Angle	-	3.97	7.76%	0.01814	0.03394	-
	Input	-	-	-	2.11766	3.00764	211.5969
75 cm	Position	1.73	7.78	23.03%	0.12001	0.26877	-
	Angle	-	3.95	7.62%	0.01772	0.03283	-
	Input	-	-	-	1.59745	2.48142	159.6030

4.4 Real-time control

Two-wheeled robots are multi variable, under actuated and naturally unstable systems. Mainly, stabilization of such systems require monitoring and control of at least two states. The first state is the tilt angle of the chassis which is used to keep the robot in upright position. The second state is the speed of the driving motors of the two-wheeled robot or its transitional speed. In this case only straight forward/backward movement will be achieved as both of the wheels will have the same speed and direction.

4.4.1 Controller design

Balancing the two-wheeled robot in the vertical position implies that the angle of the chassis with the z-axis should be zero or close to zero. In order to achieve this, a controller is needed to monitor the tilt angle and adjust the input of the driving motors. However, adjusting the motors input implies that the two-wheeled robot will be moving and its position will not be fix. Thus, another control loop is required to maintain the position of the robot while the first controller maintain the balancing state of the two-wheeled robot.

These controllers were designed as two cascaded PID based controllers, as PID controllers are simple to implement and the most common controller type used in the industry. The first controller takes the tilt angle measurement (the output of the complimentary filter) as a feedback signal, then the output of the controller is applied to the motors driver of the wheels. Hence the controller is adjusting the direction of rotation and speed of the wheels accordingly to reduce the error in the tilt angle.

Positioning and moving the robot could be achieved by choosing a non-zero set point for the first PID controller, the inner loop. A positive or negative set point will result in a forward or backward movement of the two-wheeled robot respectively. However, if a constant non-zero set point value is assigned, the controller will keep increasing the wheels speed until it violates the physical

constraints of the robot and becomes unstable; therefore, a PI controller is implemented to realize the second controller, the outer control loop. The feedback of the second controller is the displacement measurement, position of the robot, which is taken from the encoder of the wheels, while the output of the second controller is adjusting the set point of the first controller, the inner control loop according to the desired travel distance. Therefore, when the two-wheeled robot reaches the target position the output of the second controller will be zero, which is the reference input of the balancing controller. Eventually, the two-wheeled robot maintains the upright balance at the desired position. Both controllers should be working in tandem in such a way that the response of the inner control loop should reach steady sate before the feedback is taken to the outer control loop. Therefore, the sampling rate of the inner control loop is chosen to be higher than the sampling rate of the outer control loop by a factor of 10; to be 100Hz. Figure 4.13 shows the detailed configuration of the two-wheeled robot controller.

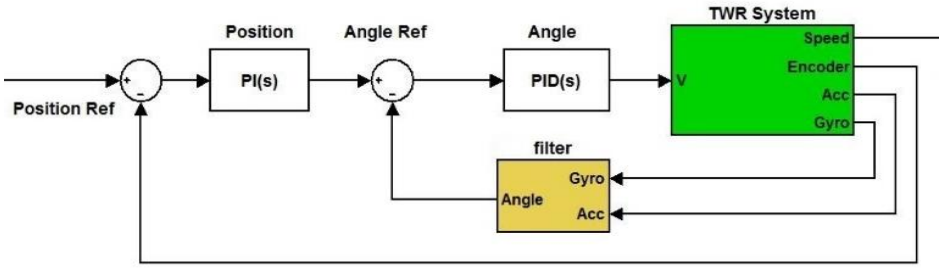


Figure 4.13: ***Two-wheeled robot controller structure***

4.4.1.1 Integral term enhancements

One of the common issues in implementing PID controllers is the integral windup in which the I-term of the controller will continue to accumulate the error even when it becomes outside of the operating range of the system. Consequently, no further change happens in the system, and the integral term of the PID keeps on growing. In such case, the system will take long time duration to return to be within the normal operating range where the controller will still respond with the integral term. This period of time is required to take the integral term down from the wound-up

condition. Although, the final output of the implemented controller has already been limited to the operating range of the motors, the I-term of the controller can build up.

Bohn and Atherton (1995) presented a couple of solutions for the integral windup problem in PID controllers one of which is limited integrator. In this simple approach the I-term output of the PID controller is checked if it is within the operating range and it will be clamped otherwise as in the following code example.

```
If ( I_term > Max_out ) I_term= Max_out;
else if ( I_term < Min_out) I_term = Min_out ;
```

Another issue in the implementation of a PID controller which is associated with the integral term of the controller is addressed in the following coding example of a PID controller. Here the I-term of the controller is calculated as Ki multiplied by the integrated error (ErrorSum). This calculation may result in an over-emphasis of the I-term and may complicate the tuning process of the controller.

```
ErrorChange = (error – OldError ) /dt;
ErrorSum = ErrorSum + (error * dt);
PID = error * Kp + ErrorSum * Ki + ErrorChange * Kd;
OldError  =  error;
```

An enhancement can be applied to fix this issue easily by replacing the (ErrorSum) calculation with an iterative calculation of the integral term and update the PID equation as in the following code lines:

```
I_term = I_term + (Ki * error * dt);
PID = error * Kp + I_term + ErrorChange * Kd;
```

4.4.1.2 Derivative term enhancements

Derivative kick is a description for a large values or spikes in the PID controller output due to derivative term in the controller. These spikes are caused by a sudden change in the set point or input of the system as described in Figure 4.14. This change leads to a big value for the error derivative which will be multiplied by the gain K_d making the spikes even bigger.

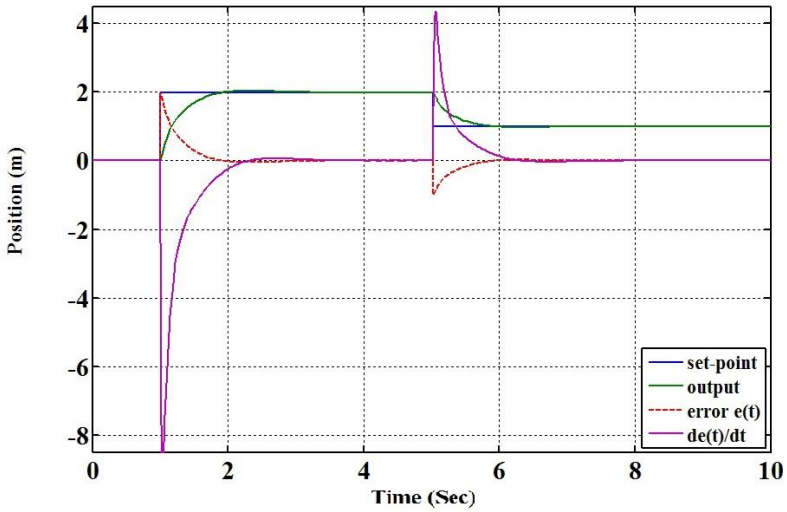


Figure 4.14: ***Derivative kick in a PID controller***

In order to eliminate the derivative kick and enhance the derivative part of the PID controller, the derivative of the error signal is replaced with the negative derivative of the input signal for the derivative term calculation, as the change in the set-point leads to an opposite change in the input signal. Thus, the implementation of a PID controller with an enhanced derivative term calculation is coded as:

```
d_Input = (Input – Old_Input)/dt ;
PID =  error * Kp + I_term – d_Input * Kd ;

Old_Input = Input;
```

4.4.1.3 Controller gains tuning

Although methodologies for tuning the parameters of the PID controller exist such as Ziegler-Nichols (Ziegler and Nichols, 1942), the controller parameters were manually tuned using a heuristic approach. The developed interactive PC software, presented in chapter 2, is utilized during the tuning process. Subsequently the controller parameters i.e. error, integral error etc. are monitored online whilst updating the gains value to find a suitable combination which provides satisfactory good results. Figures 4.15 and 4.16 show the experimental results for the chosen gain values, as noticed the controller is maintained the tilt angle of the robot effectively within ±1º.

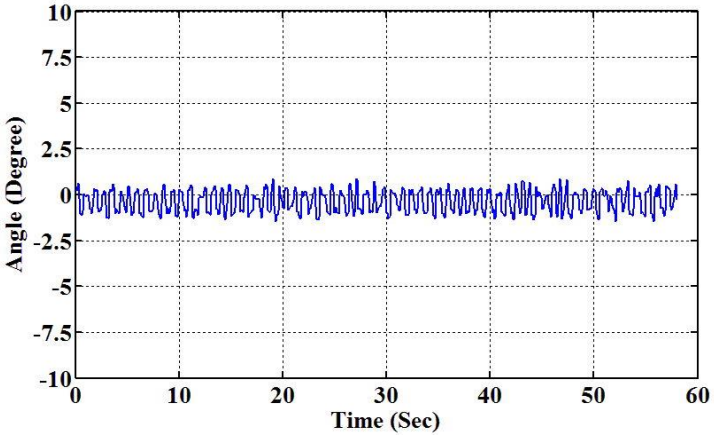


Figure 4.15: ***Tile angle measurement of the robot***

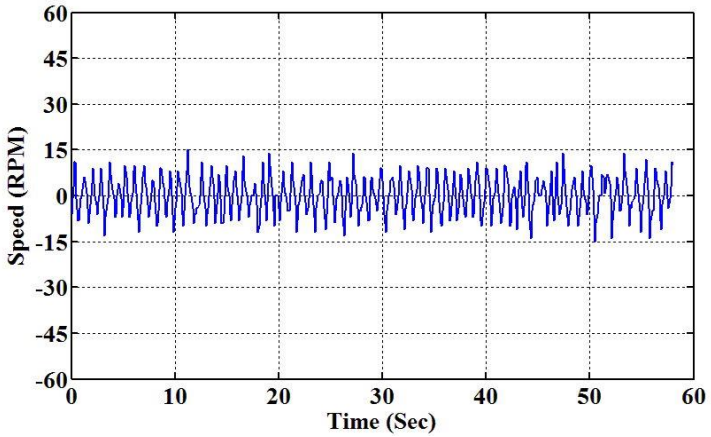


Figure 4.16: ***Speed measurement of the robot's wheels***

4.4.2 Steering control

The steering of the robot can be achieved by setting different input values to the left and right motor. However, this will make the controller of the robot more complicated and more difficult to tune as there will be a controller for each wheel. Therefore, a differential steering mechanism is implemented, where a steering command is added to the applied control signal of the right wheel and subtracted from the control signal of the left wheel at the same time as shown in Figure 4.17.

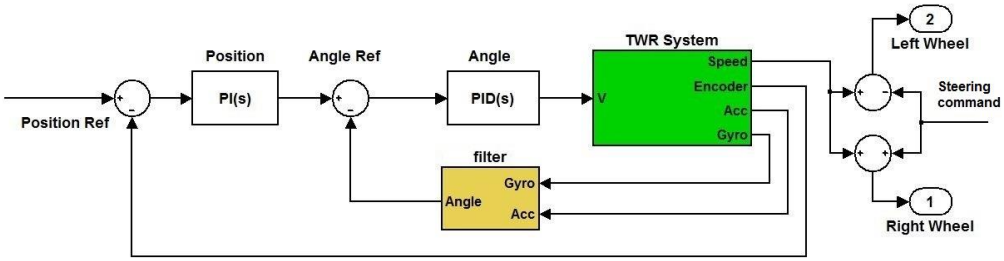


Figure 4.17: ***Differential steering mechanism of the robot***

This mechanism will provide multiple steering modes and maintain the average speed of both wheels during balancing. For instance, if the steering command input is zero then the two-wheeled robot will be either in straight movement mode or balancing mode based on the travel speed set point. Whereas, a non-zero steering command input with travel speed set point of zero the two-wheeled robot will rotate and spin around its vertical axis. The robot will turn around one of its wheels, corner turn, for an equal input value of both steering command input and travel speed reference. On the other hand, the two-wheeled robot will follow a curve path for a non-zero not equal values of steering command input and travel speed reference. Table 4. defines the motion behaviour of the two-wheeled robot for different values of travelling speed reference and steering command input.

Table 4.3: ***Motion behaviour of the robot***

Travel speed ref	Steering command ref	Robot's motion behaviour
0	0	Balancing
+Ve	0	Forward
-Ne	0	Backward
0	+Ve / -Ne	CW/CCW rotate around its vertical axis
+Ve / -Ne	+Ve / -Ne	CW/CCW arc turn movement

In order to test the various motion behaviours of the two-wheeled robot, a forward backward movement test was conducted. In this test a command is sent to the robot to move two metres forward at wheels speed of 15 RPM then move backward to the original start position at the same speed. Figures 4.18 and 4.19 show the performances of the robot in terms of tilt angle and wheels speed during the test while Figure 4.20 shows the travelled distance of the robot. It is clearly noticed that the travelled distance result is not smooth as the balancing controller adjusts the speed of the wheels to maintain the upright balance.

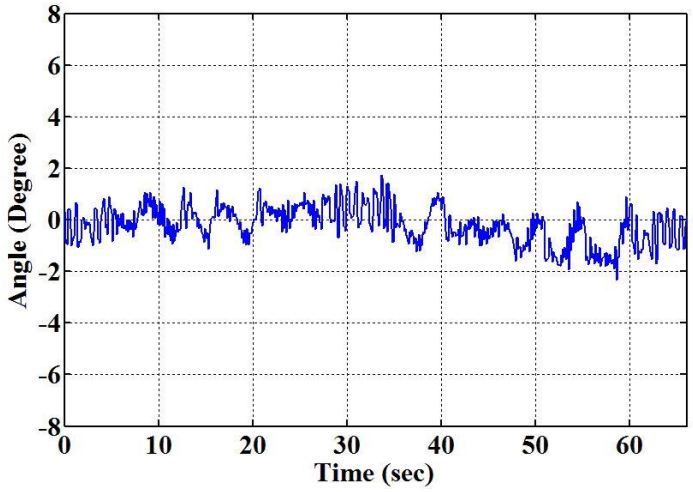


Figure 4.18: ***Tilt angle measurement of the robot moving forward and backward***

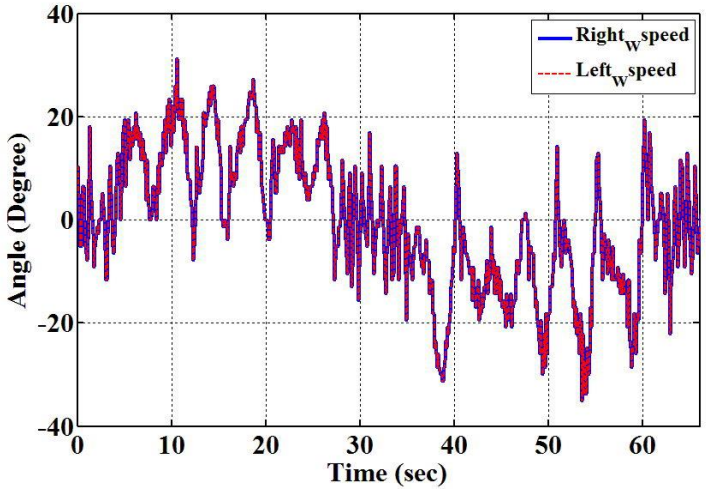


Figure 4.19: ***Speed measurement of the robot's wheels moving forward and backward***

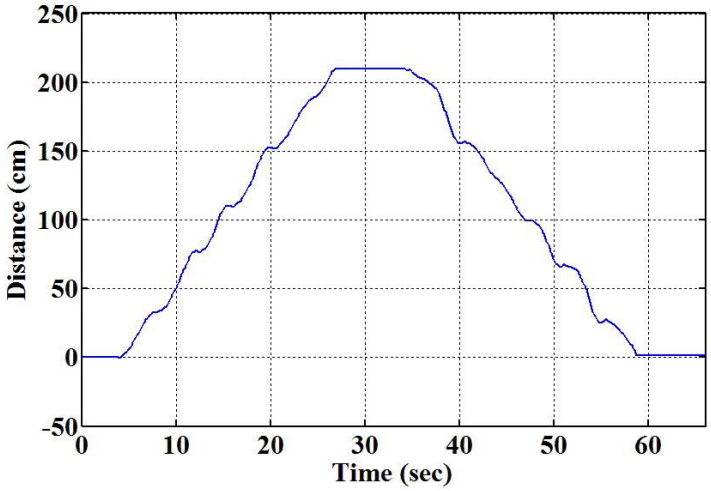


Figure 4.20: ***Position of the robot's wheels moving forward and backward***

Steering of the robot was tested for two steering modes turn around corner, one of the wheels fixed, and turning into a curve path. The turn around a corner mode was tested by setting an equal value of +5 RPM for the travel speed reference and the steering command input. Hence the robot turned around its left wheel as the average speed of the left wheel will be zero and the right wheel is moving forward with a speed of +10 RPM according to the differential steering. Figures 4.21 to 4.23 depict the experimental results of the steering test in terms of the tilt angle, the speed of the wheels, and the position of the robot.

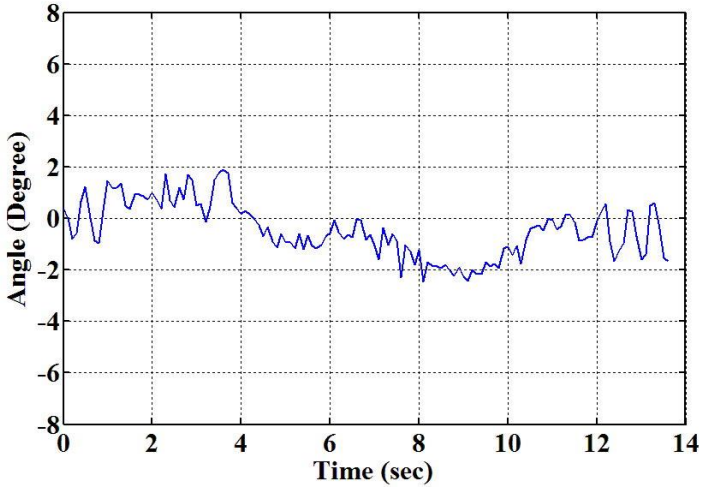


Figure 4.21: ***Tilt angle of the robot turning left corner***

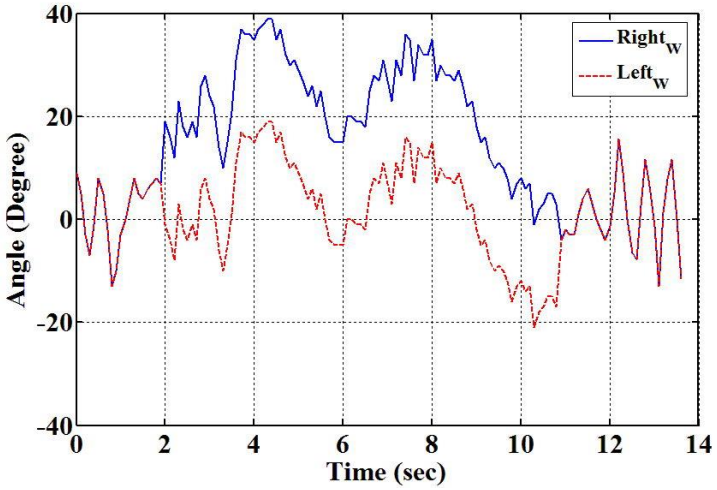


Figure 4.22: ***Speed of the robot's wheels turning left corner***

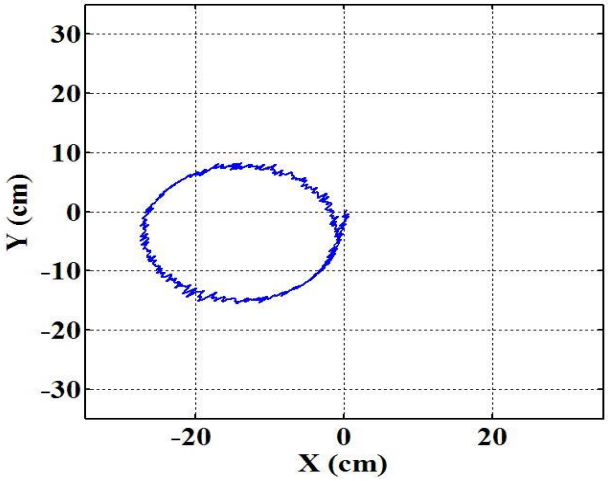


Figure 4.23: ***Position of the robot during turning around its left wheel***

Similarly, the turn in the opposite direction test is conducted by setting the steering command input to -5 and keeping the travel speed set point unchanged. Therefore, the robot turned around its right wheel as the average speed of this wheel will be zero. Figures 4.24 to 4.27 depict the experimental results of the steering test in terms of the tilt angle, the speed of the wheels, and the position of the robot.

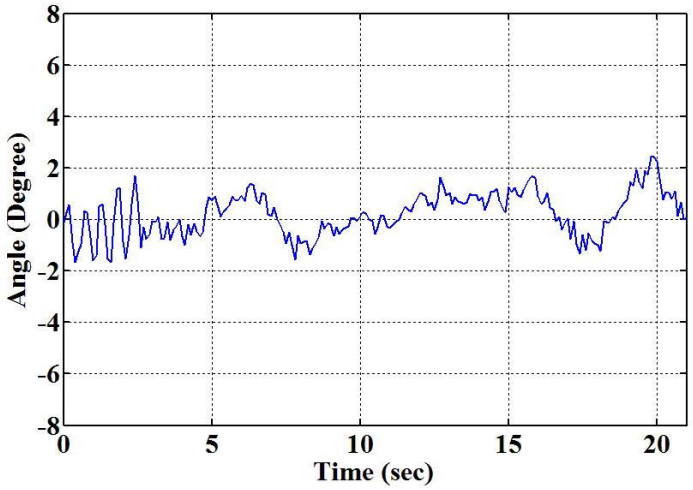


Figure 4.24: ***Tilt angle of the robot turning right corner***

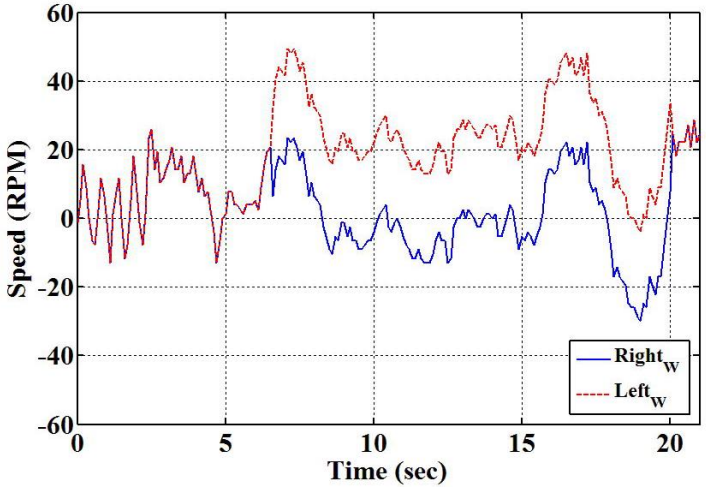


Figure 4.25: ***Speed of the robot's wheels turning right corner***

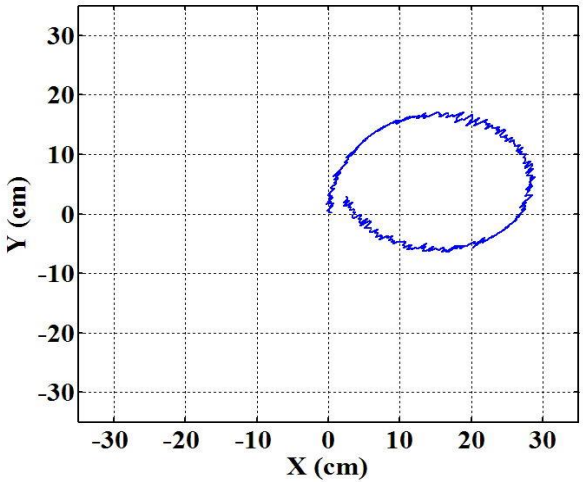


Figure 4.26: ***Position of the robot during turning around its right wheel***

The second steering mode of the robot, curve turning, was evaluated by setting a non-equal value for the travel speed set point and the steering command input. Therefore, none of the wheels will have an average speed of zero and the two-wheeled robot will move into a curve path bending to the left or right based on which wheel has the higher average speed. The experiment was conducted with travel speed reference of 10 RPM and steering command input of 5, so that the average speed of the right wheel will be higher than the average speed of the left wheel. Figures 4.27 to 4.29 show the performance of the two-wheeled robot in terms of the tilt angle, the speed of the wheels, and the position of the robot respectively. As noticed from Figure 4.29 that the robot is moved into a curved path turning right, counter clock wise (CCW).

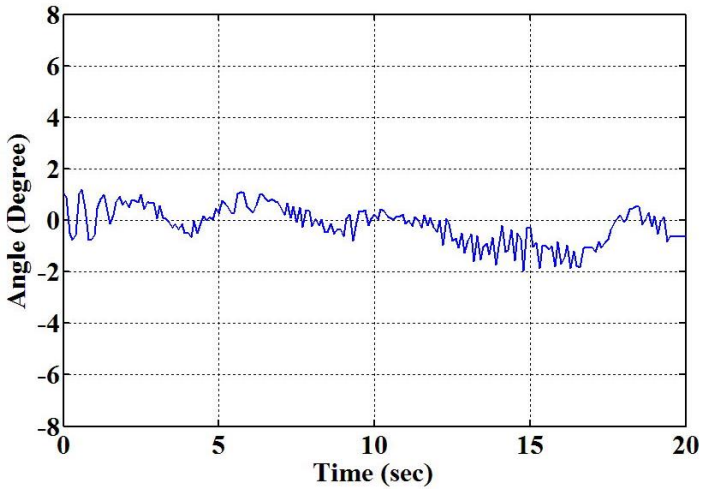


Figure 4.27: ***Tilt angle of the robot turning left curve***

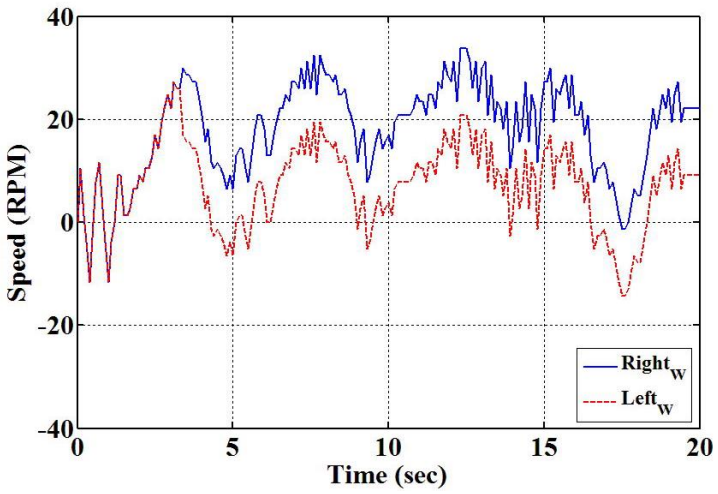


Figure 4.28: ***Speed of the robot's wheels turning left curve***

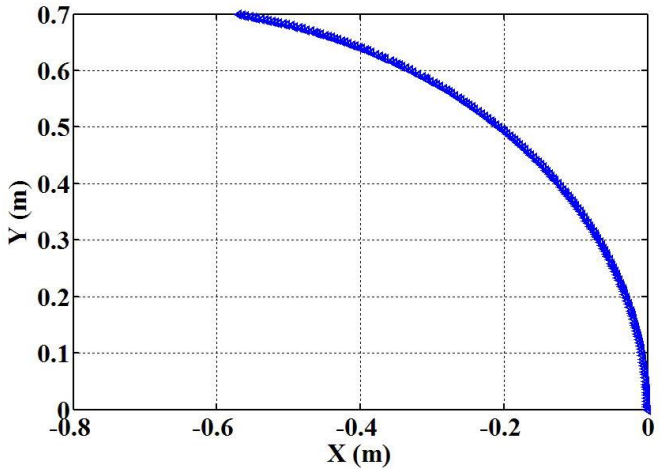


Figure 4.29: ***Trajectory of CCW curve motion of the robot rotating***

The final motion behaviour of the two-wheeled robot, spinning or zero radius turn, was evaluated by setting the travel speed set point to zero and the steering command input to – 5, so that the speed of the left and right wheels of the robot will have the same value but in opposite direction according to the differential steering. Therefore, the robot will rotate / spin around its vertical axis in CCW. Figures 4.30 to 4.32 show the performance of the robot in terms of tilt angle, speed of the wheels, and position of the robot respectively.

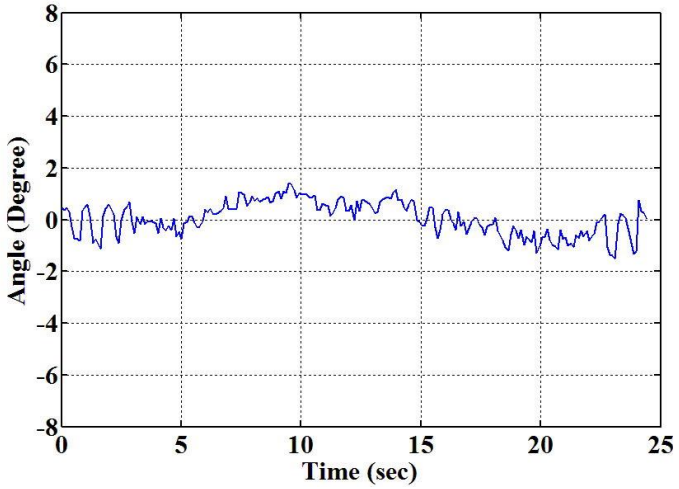


Figure 4.30: ***Tilt angle of the robot rotating with turn input of -5***

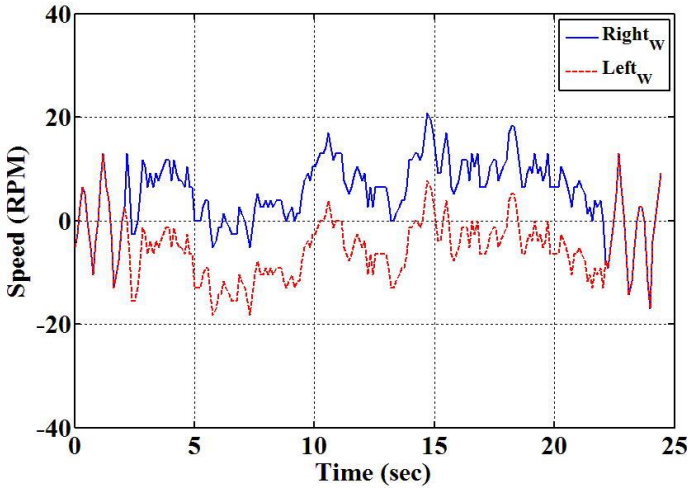


Figure 4.31: ***Speed of the robot's wheels rotating with turn input of -5***

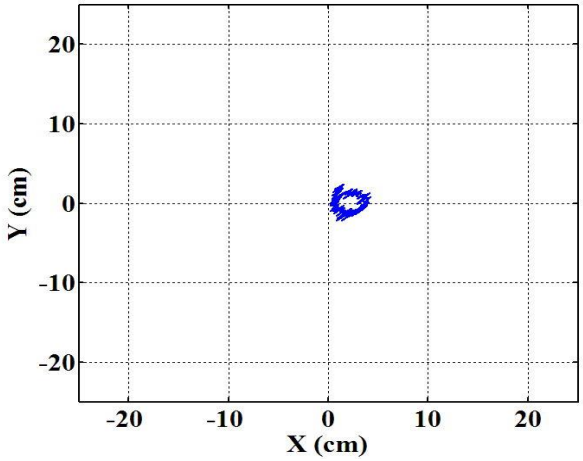


Figure 4.32: ***Position of the two-wheeled robot rotating with turn input of -5***

However, for a bigger steering command input value, a higher speed of rotation, such as -13, the two-wheeled robot will lose its balance and collapse due to the generated centrifugal force. Figures 4.33 to 4.35 depict the experimental results of spinning with relatively big turn input in terms of the tilt angle, the speed of the wheels and the position of the robot.

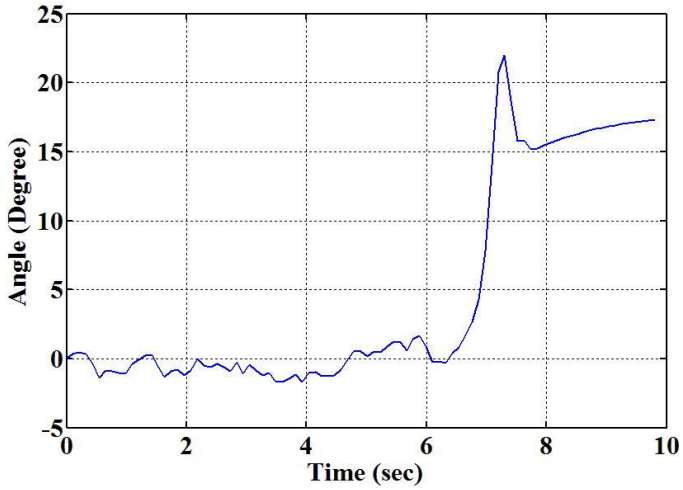


Figure 4.33: ***Tilt angle of the robot rotating with turn input of -13***

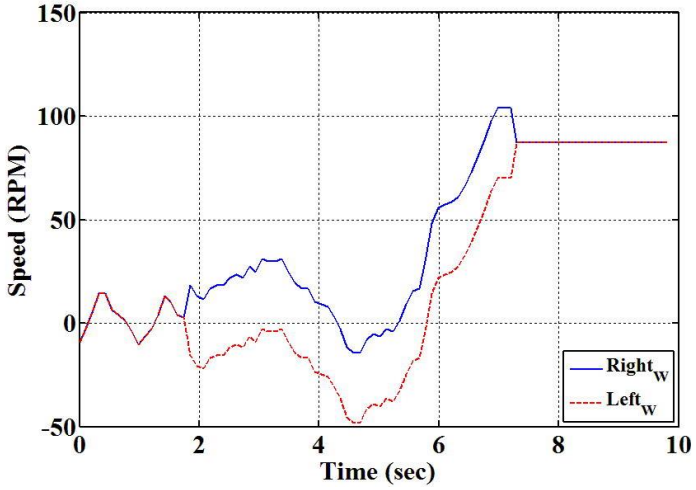


Figure 4.34: ***Speed of the robot's wheels rotating with turn input of -13***

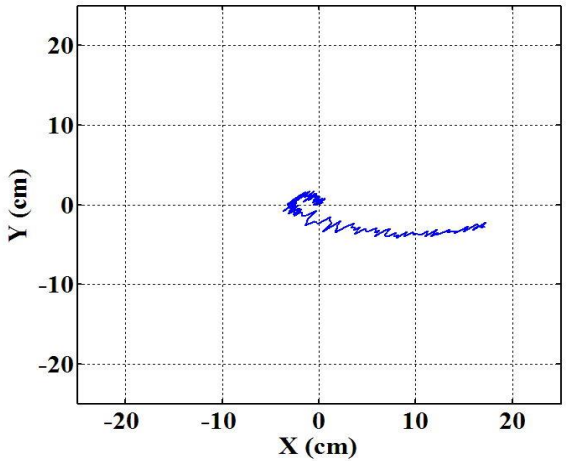


Figure 4.35: ***Position of the two-wheeled robot rotating with turn input of -13***

4.4.3 Self-erecting control

This term refers to the ability of the two-wheeled robot to stand up and balance upright from the resting position, rest on one of its auxiliary casters, without human intervention. The main challenge is when the controller tries to balance the two-wheeled robot from an initial position, tilt angle, far away from the set-point; the error will be too big and will result in large unacceptable overshoot. Consequently, leading to large and oscillatory wheel speeds and eventually driving the robot outside of the safety boundaries. Therefore, a dual-mode approach for the balancing controller is considered. The first mode, self-erecting mode, is activated first and it is responsible for reducing the error of the tilt angle to a manageable value, then switch to the second mode, balancing mode, that enables normal balancing and keeps the error of the tilt angle close to zero.

The self-erecting controller structure is shown in Figure 4.36. The concept is to modify the balancing controller by altering its characteristics to act as a self-erecting mode which is responsible for lifting the two-wheeled robot from its rest position. In fact, only a change in the proportional gain is found to be sufficient. Reducing this gain effectively reduces the overshoot and drives the robot into a safe region. The two-wheeled robot is able to maintain stability with the reduced proportional gain, however the performance is degraded; therefore the original parameters are restored after a certain time in order to activate the balancing mode which resumes the normal balancing behaviour of the robot. By analysing the experimental result, it was found that activating the self-erecting mode for 2-3 seconds is sufficient to bring the two-wheeled robot into upright balance.

The performance of the controller was tested when the two-wheeled robot was resting on one of its auxiliary casters and causing a positive initial tilt angle of about 6 degrees. The experimental results in Figures 4.37 and 4.38 demonstrate the system performance in terms of the tilt angle and the speed of the wheels.

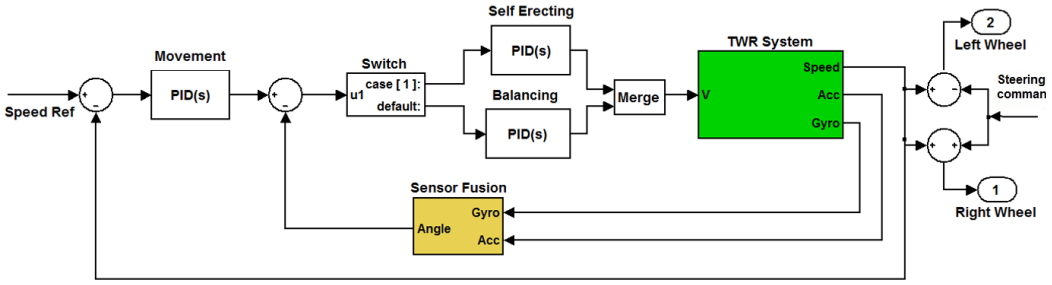


Figure 4.36: ***The Self-erecting controller structure***

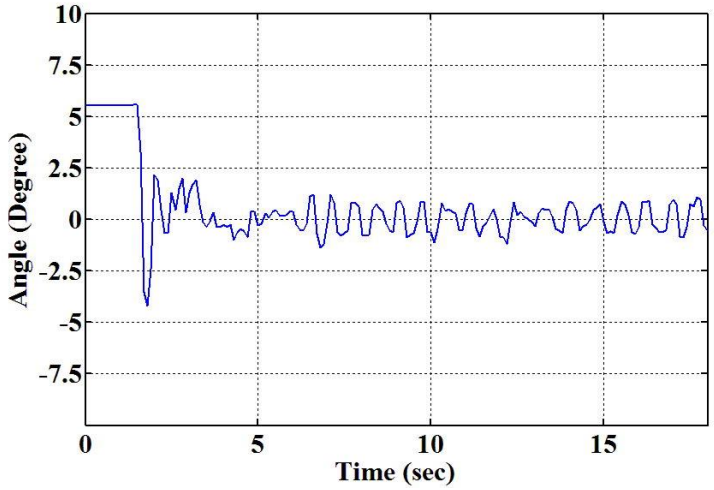


Figure 4.37: ***Tilt angle for self-erecting from a +Ve initial angle***

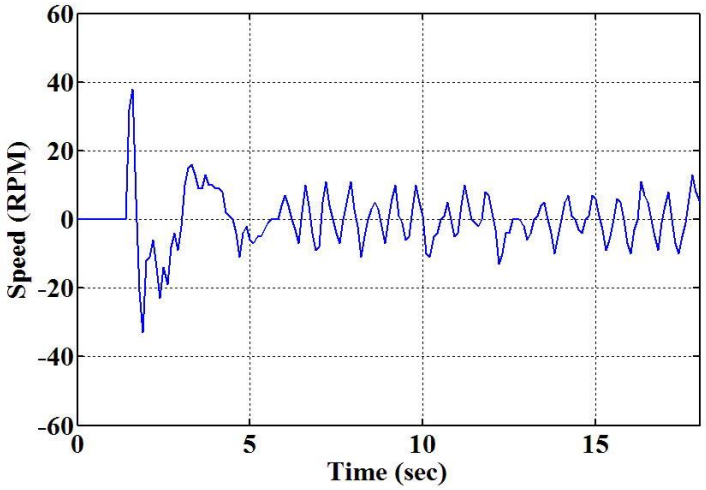


Figure 4.38: ***Wheels speed for self-erecting from a +Ve initial angle***

Another test was conducted whilst the two-wheeled robot was resting on the other side that generates a negative tilt angle of 7.5 degrees for the z-axis. Again, the controller is successfully maintained the balance of the robot as demonstrated in Figures 4.39 and 4.40.

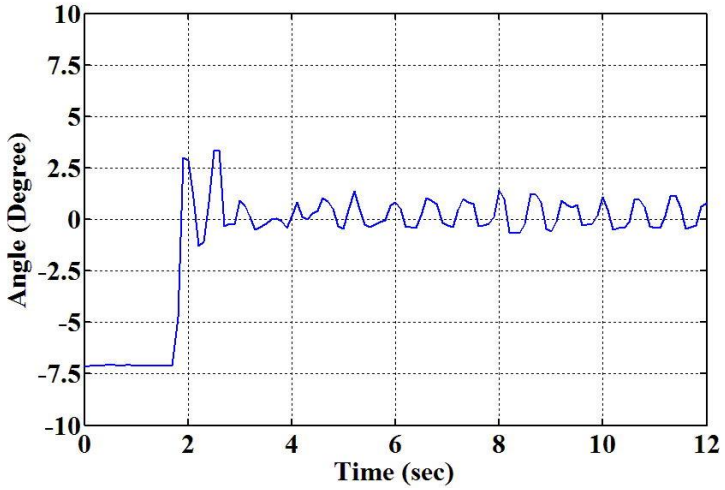


Figure 4.39: ***Tilt angle for self-erecting from a -Ve initial angle***

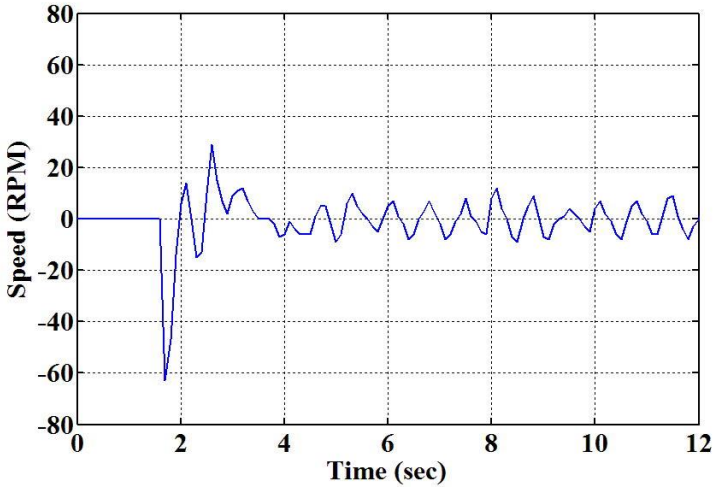


Figure 4.40: ***Wheels speed for self-erecting from a -Ve initial angle***

4.5 Development of the two-wheeled robot controller

The control strategy for the two-wheeled robot was enhanced and the controller structure was modified to accommodate the new modification on the robot hardware. These modifications include adding a wheels synchronizer, another control loop for the payload actuator, and a new controller structure for balancing and positioning the two-wheel robot. These will be discussed in the following sections.

4.5.1 Wheels synchronizer

After installing and testing the new motors of the robot, a small issue was identified in motors synchronization. Ideally, the rotation speed of both motor's shaft should be the same if the same control signal is fed to the motors, but in practice the motors did not have the same speed. Although the motors are the same type from the same manufacturer and driven by the same input, there was a slight difference in the rotation speed which affects movement of the robot on a straight line. The reasons for this difference could be related to a mismatch in the motor parameters such as coil resistance, magnetic field, friction, and gears.

In order to solve the mismatch issue, a method for online motors speed synchronization was developed. A simple PI controller was designed and realized with the block diagram given in Figure 4.41 as synchronization controller. This controller adjusts the input signals of the motors aiming to maintain zero difference between the left and right encoder. This method of synchronization would be better than scaling on of the motor's input as it may be changed by time. Figure 4.42 shows the experimental results of wheel travel distances with and without wheel synchronization. It is clearly noted that wheel synchronization controller was effective to solve the mismatch problem.

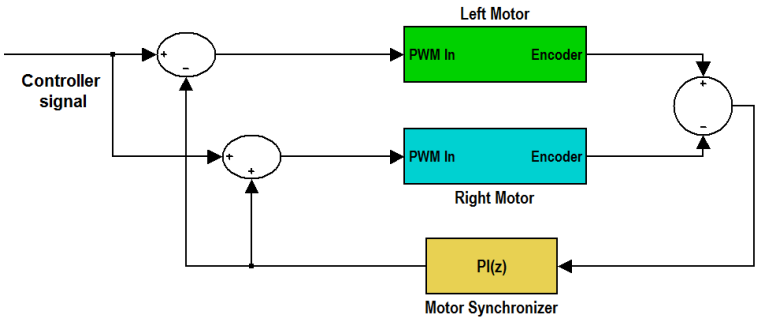
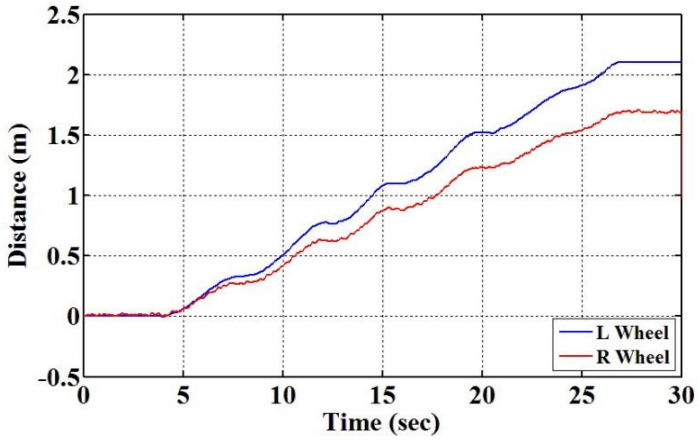
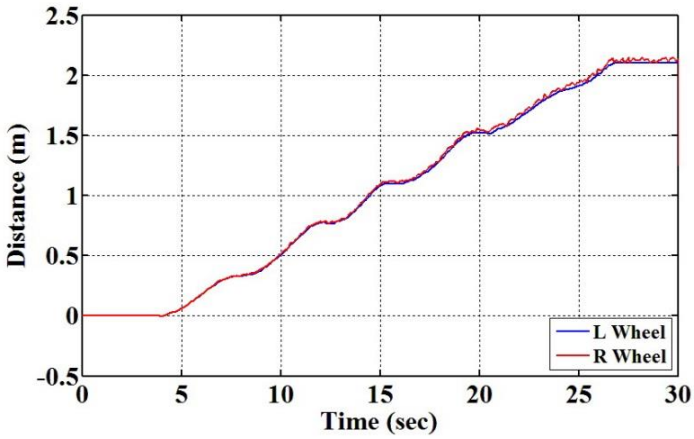


Figure 4.41: ***Wheel synchronizer block diagram***



(a)



(b)

Figure 4.42: ***The response of the wheel synchronizer***

a) ***Without wheel synchronizer*** ***b) With wheel synchronizer***

4.5.2 Design of payload actuator controller

The movement of the payload was controlled through a stepper linear actuator. A controller was developed to take the payload to a certain position up and down along the chassis of the two-wheeled robot as in Figure 4.43. The controller represents a simple stepper motor position controller that takes the position reference and the direction of the movement from the interface PC software. Through this software, the user can activate, deactivate, and set the direction of the payload movement. The payload controller was tested for different positions at different times as illustrated in Figure 4.44.

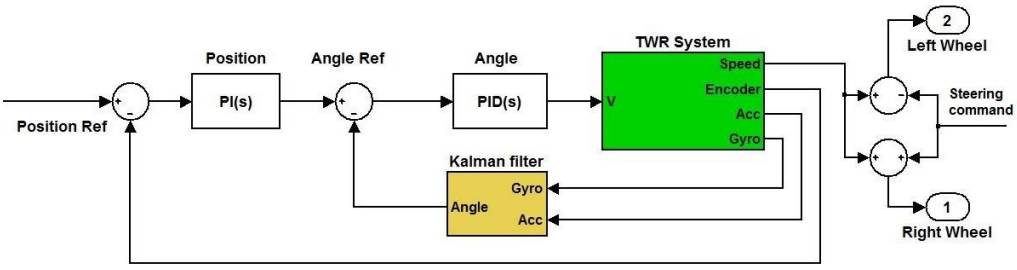


Figure 4.43: ***The controller structure for the moving payload***

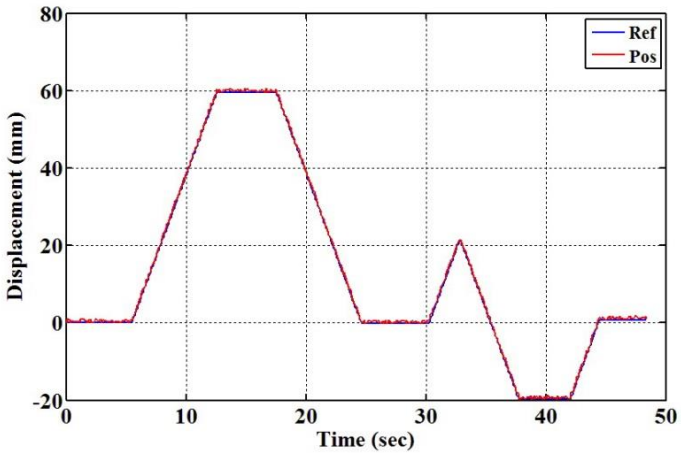


Figure 4.44: ***Example of the payload movement***

4.5.3 New controller structure

New controller for balancing and positioning the two-wheel robot was developed. It consists of three control loops as in Figure 4.45. The first loop, represented by a position controller for the payload position, while the other loops represented by a PID based control loop. This loop is responsible for balancing the robot, in cascade with a hybrid controller for position of the robot. The balance controller, the inner control loop, was realized by classic PID controller which takes the Kalman tilt angle measurement as feedback and generates the control signal for the motors of the robot. Movement of the two-wheeled robot was initiated by setting an offset to tilt angle reference. The output of the positon controller was considered as tilt reference input for the balance controller, similar to the previous controller design discussed in chapter four. However, the structure of the position controller is different in this design. The position controller, the outer control loop, takes the position and the speed of the robot as inputs. It was divided into two parts as described in Figure 4.46.

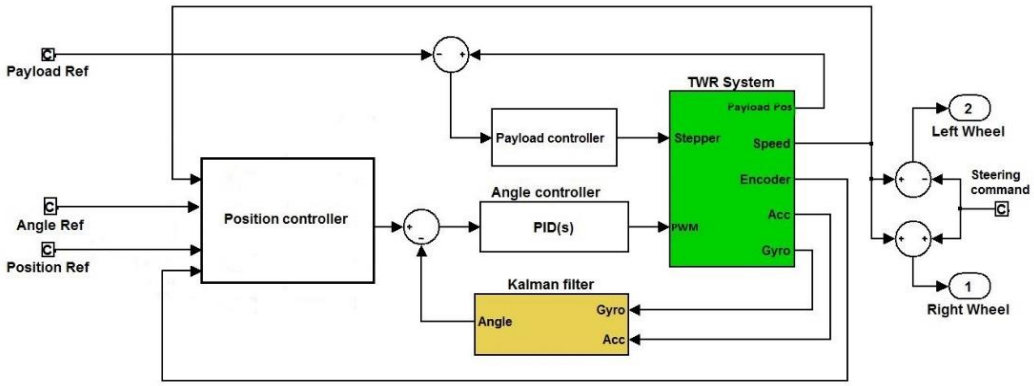


Figure 4.45: ***The new structure of the two-wheel robot controller***

The first part of the position controller was designed to govern the position of the two-wheeled robot in balancing mode and when it stops moving / manoeuvring. Hence, it will be activated only in these modes. This part takes the positon of the robot as feedback in order to generate an offset for the tilt reference of the balance controller, the inner control loop. In the meantime, the speed of the robot was used in feedforward configuration to scale that value and to add it to the tilt offset. In order to ensure smooth deceleration and fast braking/stopping of the two-wheeled

robot, gain scheduling approach was utilized with three different gain values. Every gain was assigned to a predefined position error zone. The controller will automatically switch between these gains (Z1, Z2, and Z3) according to the position error value.

The second part of the position controller was developed to adjust the speed of the robot, therefore it will be activated when the robot is moving. This part takes the speed of the robot only as feedforward input and scales it to generate the tilt angle offset, which is used to calculate the tilt reference input of the balance controller.

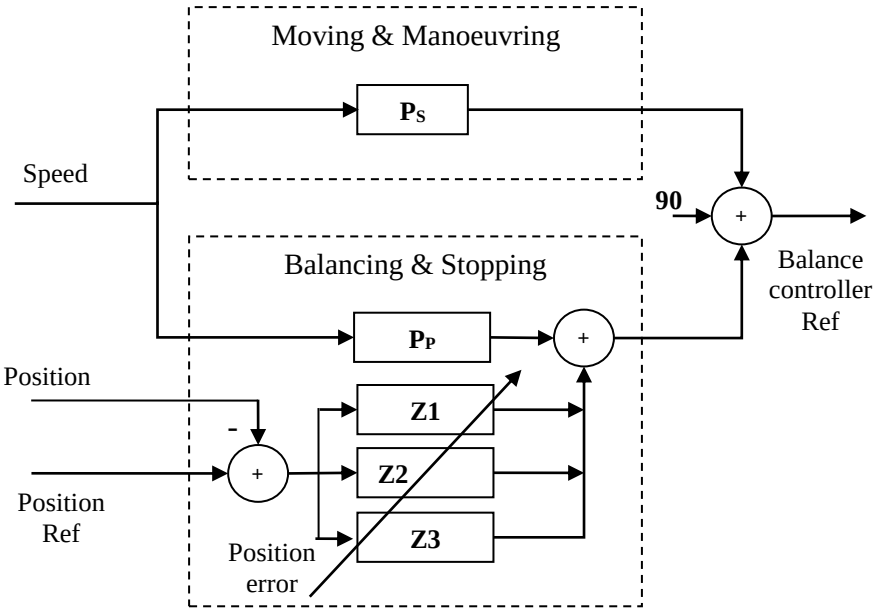


Figure 4.46: ***The internal structure of the position controller***

4.6 Real time experimental results

The developed control loops of the two-wheeled robot were digitally coded using C++ code and implemented on the Mbed microcontroller for real-time execution. The sampling time of the system was chosen as 10 msec for the balance controller, inner control loop, and 100 msec for the position and speed controller in order to keep the synchronization between the two loops. These times were found to be adequate to follow the robot dynamics.

Then, the PC software was used in the tuning process of the controller in order to choose the best possible gain values through a heuristic approach. In order to evaluate the performance of the developed control algorithm of the two-wheeled robot, two of experiments were conducted under various test conditions.

4.6.1 Fixed payload test

Balancing test was conducted with a 0.2 kg payload at a fixed position at the COM of the robot. Figures 4.47 to 4.49 show the results of the experiment in terms of the tilt angle, the normalized speed of the wheels, and the error in the position of the robot. The balance controller of the robot achieved the goal effectively keeping the tilt angle of the robot within ±0.5º. At the same time the position controller kept successfully kept the two-wheeled robot at fixed position within acceptable error margin as noted in Figure 4.49.

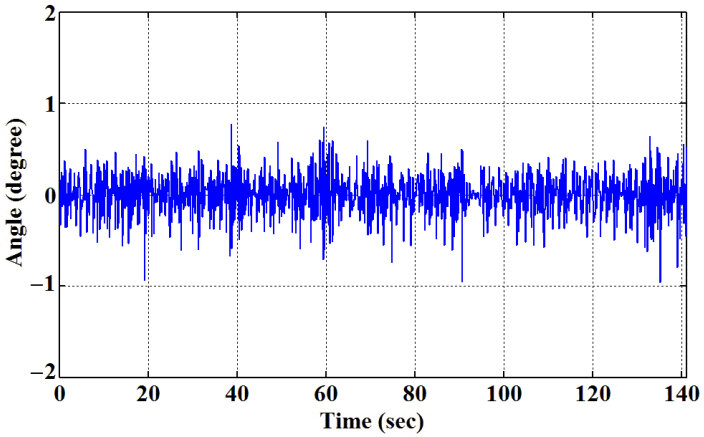


Figure 4.47: ***Tilt angle measurements while balancing with a fixed payload position***

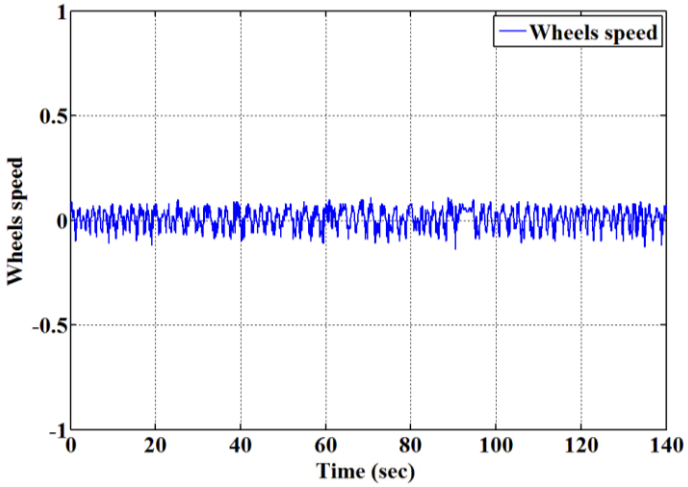


Figure 4.48: ***Normalized wheels speed while balancing with a fixed payload position***

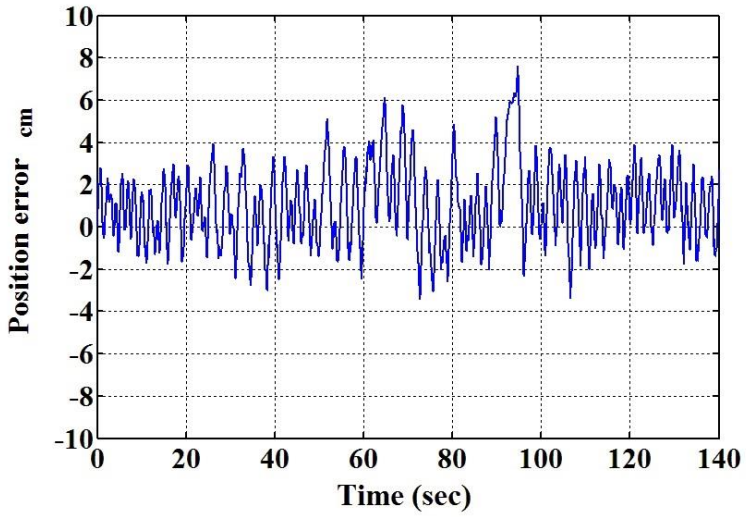


Figure 4.49: ***Position error while balancing with a fixed payload position***

4.6.2 Moving payload test

After testing the two-wheeled robot with a fixed position of the payload, a second test was conducted, which involved changing the position of 0.2 kg payload to different heights whilst the robot in its balancing state. The movement profile of the payload is shown in Figure 4.50. Whereas, Figures 4.51 to 4.53 show the performance of the robot in terms of the tilt angle, the normalized wheels' speed, and the position error whilst moving the payload. It is noted that the deflection in the tilt angle was within ±1°.

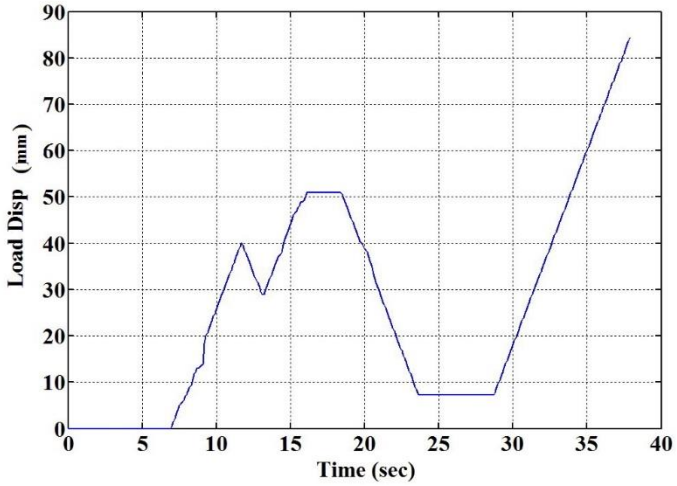


Figure 4.50: ***Displacement profile of the payload position***

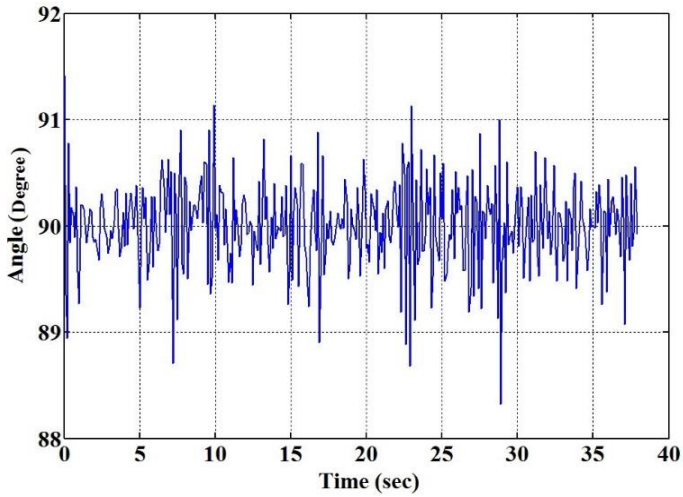


Figure 4.51: ***Tilt angle measurements in balancing state with moving payload***

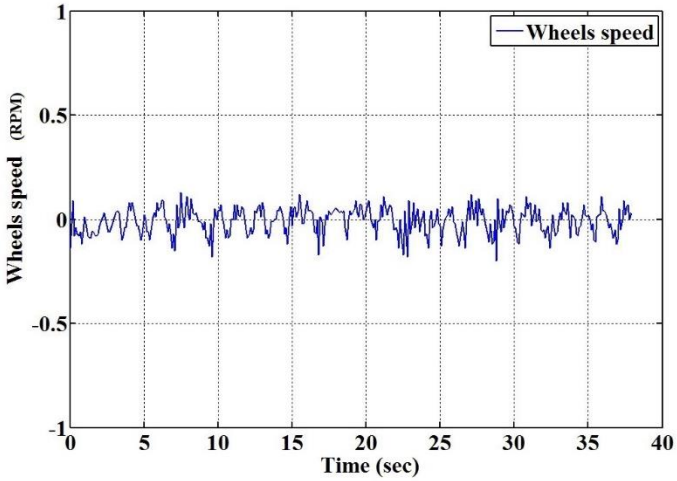


Figure 4.52: ***Normalized wheels speed in balancing state with moving payload***

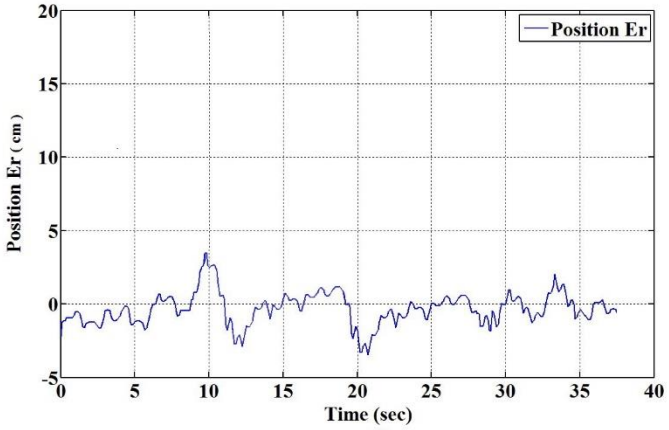


Figure 4.53: ***Position error of the robot with moving payload***

Two more scenarios were considered with two different payload weights of 350g and 500g. The experiment involved changing the payload position dynamically to different heights whilst the robot was in its balancing state. In the first scenario, the payload of 350g was lifted upward by about 18 cm from the robot's COM in two steps. This motion was made according to a predefined profile as shown in Figure 4.54. The performance of the system is depicted in Figures 4.55 to 4.57 in terms of the tilt angle, position error, and the normalized motors' input respectively.

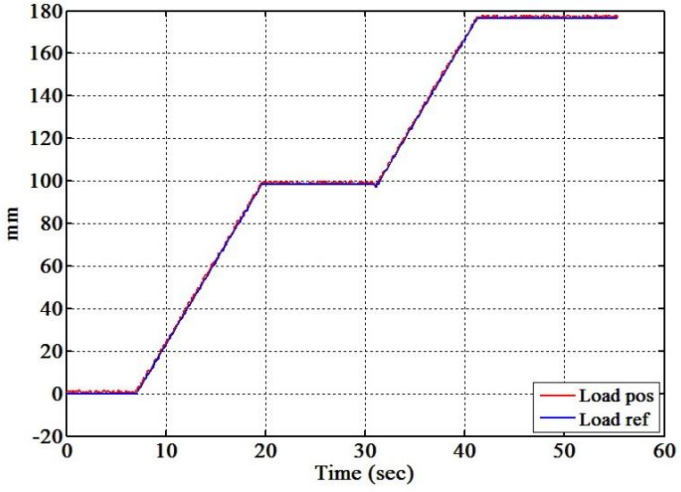


Figure 4.54: ***Displacement profile of a 350g payload position***

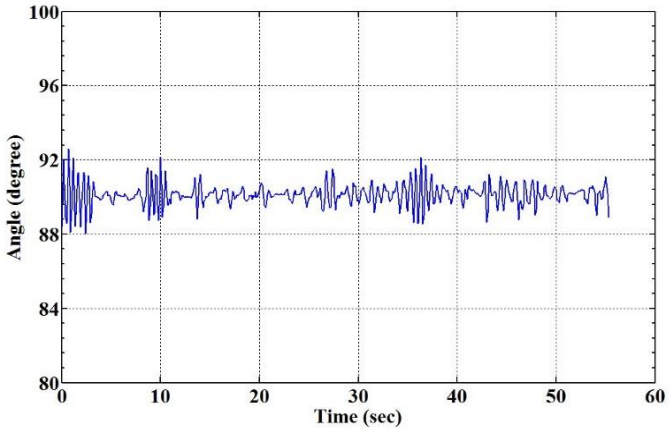


Figure 4.55: ***Tilt angle measurements in balancing state with 350g payload moving up***

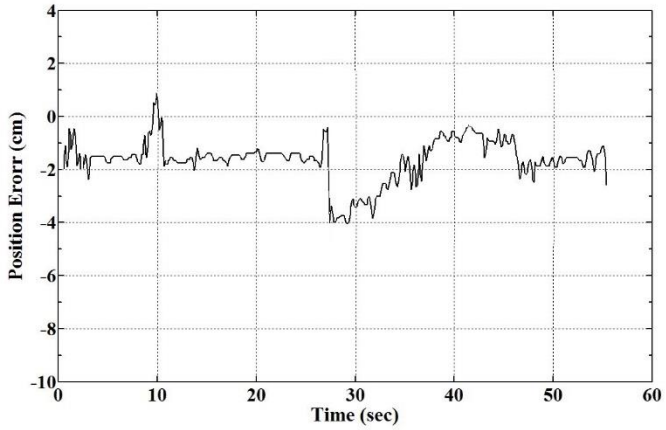


Figure 4.56: ***Position error of the robot with 350g payload moving up***

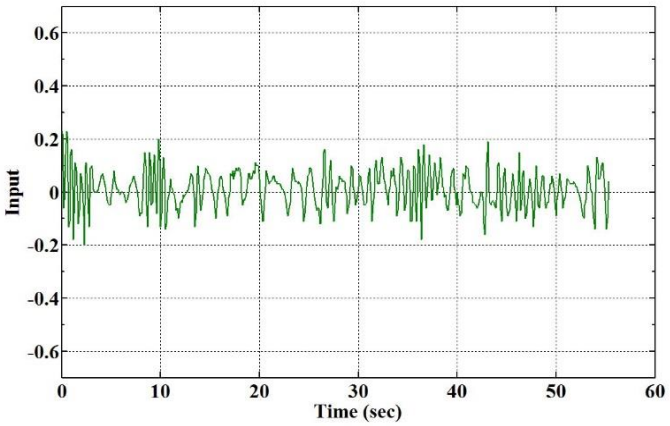


Figure 4.57: ***Normalized input of the robot with 350g payload moving up***

Then, the 350g payload was moved down by 16cm to a position close to the original COM of the robot. Figure 4.58 illustrates the predefined profile of the payload movement. The response of the robot to the payload movement is presented in Figures 4.59 to 4.61 in terms of the tilt angle of the chassis, the error in the position of the robot, and the normalized speed of the wheels, which reflects the controller efforts.

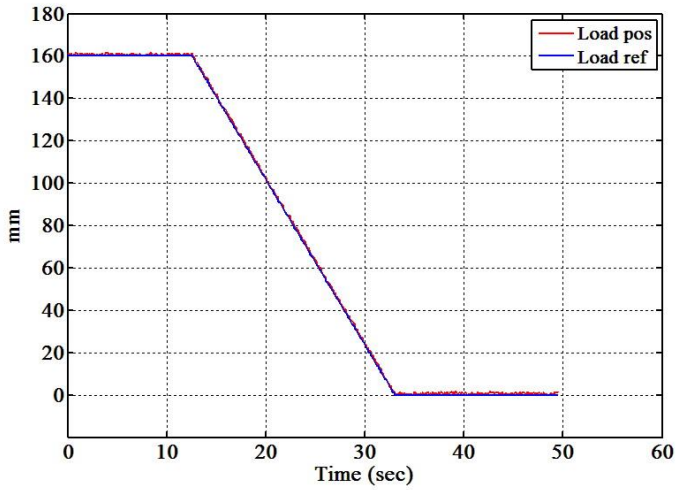


Figure 4.58: ***Displacement profile of 350g payload position***

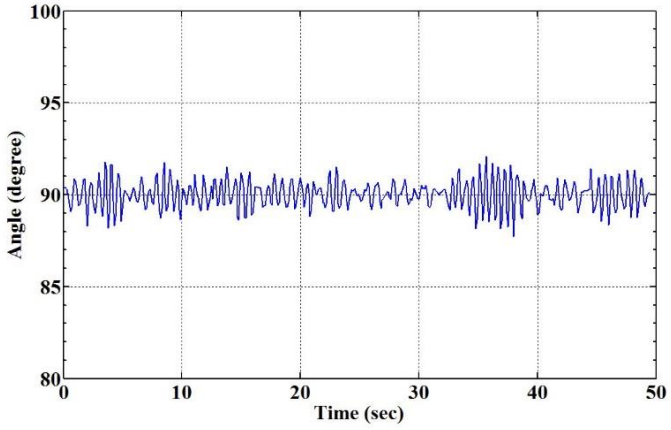


Figure 4.59: ***Tilt angle measurements in balancing state with moving down 350g payload***

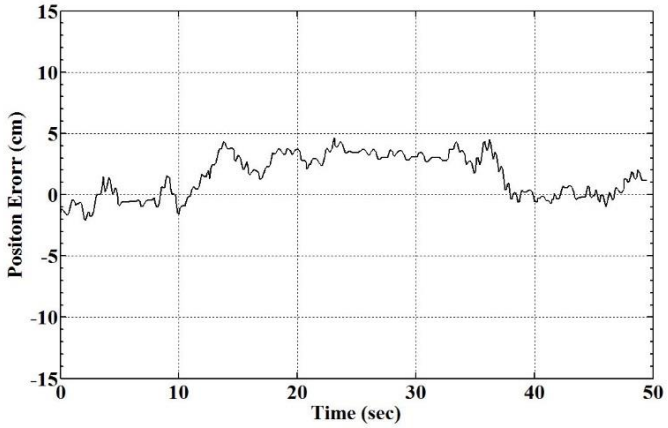


Figure 4.60: ***Position error of the robot with moving down 350g payload***

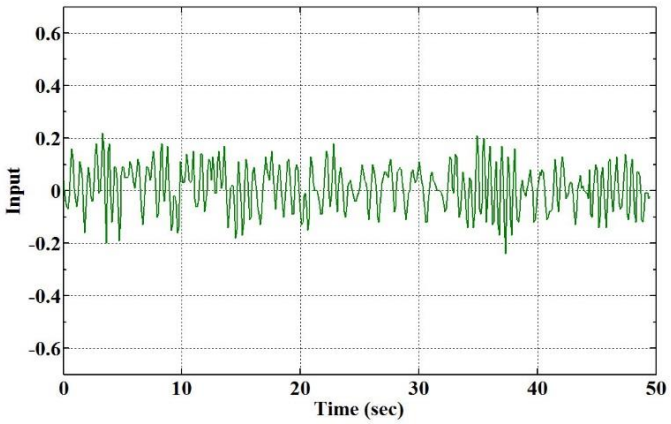


Figure 4.61: ***Normalized input of the robot with moving down 350g payload***

The tilt angle experimental results proved that the developed controller has successfully kept the two-wheeled robot balanced within an acceptable tilt angle range during the movement of the payload. Also the controller has maintained the initial position of the robot despite of small variation in the position of the robot.

In another test, the payload weight was increased to 500g and the experiment was repeated for both up and down motion scenarios. At the beginning of the test, a predefined motion profile was set as in Figure 4.62 in order to move the payload 16cm upward form its initial position at the middle layer of the robot. The response of the two-wheeled robot with the payload moving up was recorded and plotted in Figures 4.63 to 4.65.

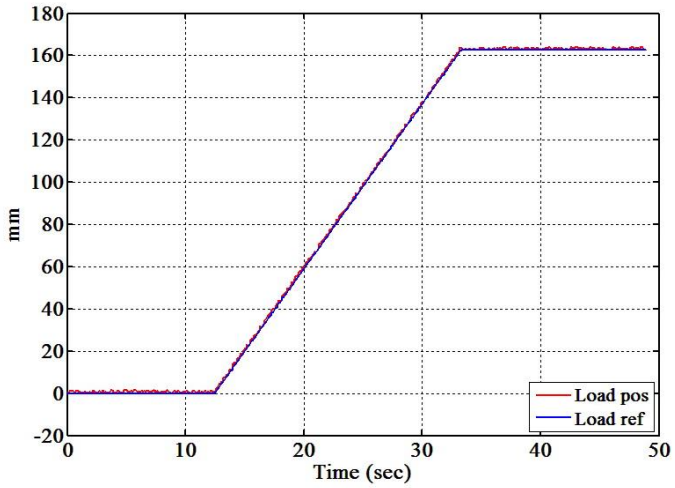


Figure 4.62: ***Displacement profile of a 500g payload position***

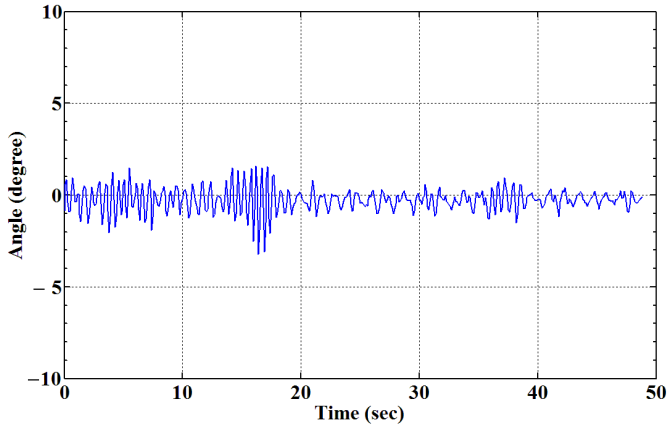


Figure 4.63: ***Tilt angle measurements in balancing state with 500g payload moving up***

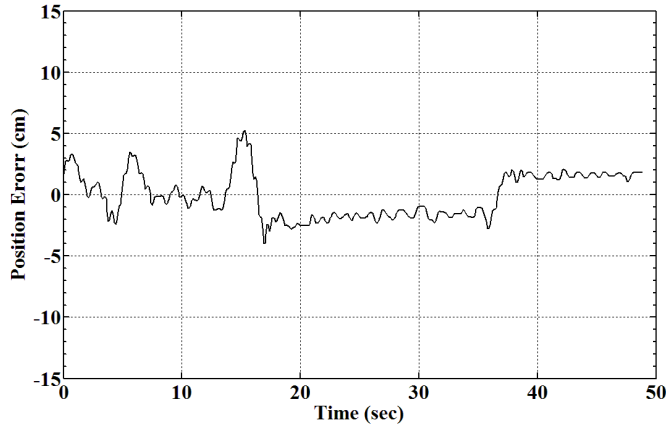


Figure 4.64: ***Position error of the robot with 500g payload moving up***

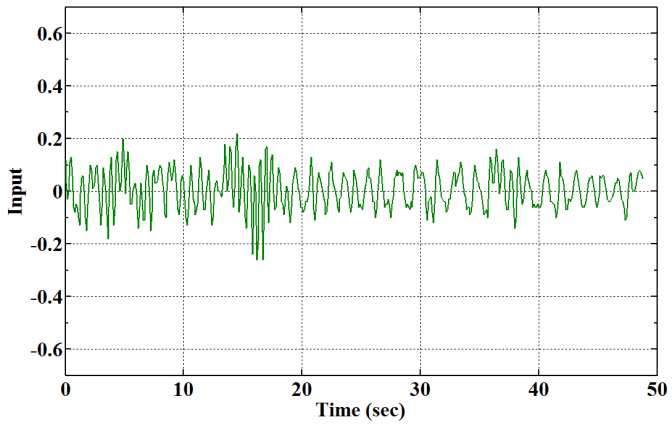


Figure 4.65: ***Normalized input of the robot with 500g payload moving up***

In the second scenario the 500g payload moved downward from its previous position in three steps. In the first two steps the payload moved by about 6.5 cm each, then by 3 cm in the last step to move the payload back to its initial position. Figure 4.66 shows the motion profile of the payload. The payload was kept at its position for 10-15 sec after each step in order to give enough settling time after each change. The response of the two-wheeled robot with the payload moving down was recorded and plotted in Figures 4.67 to 4.69.

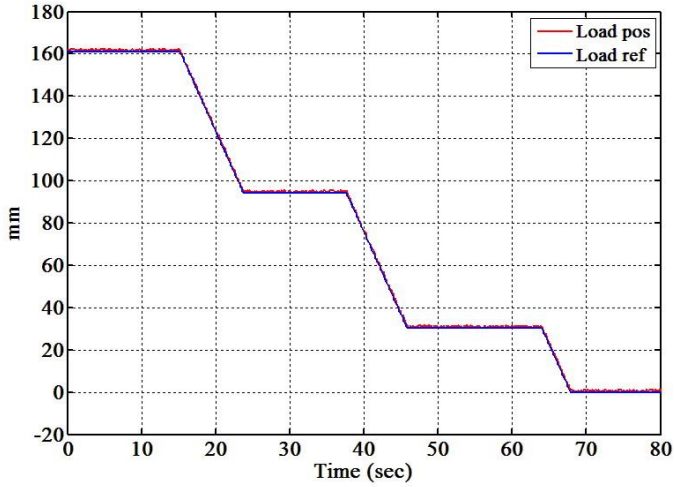


Figure 4.66: ***Displacement profile of the payload position***

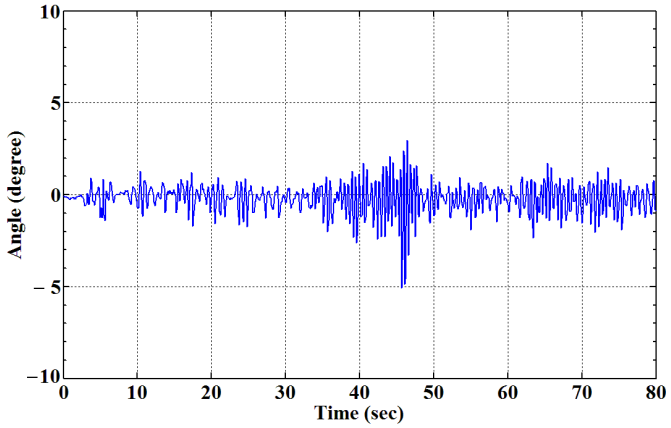


Figure 4.67: ***Tilt angle measurements in balancing state with 500g payload moving down***

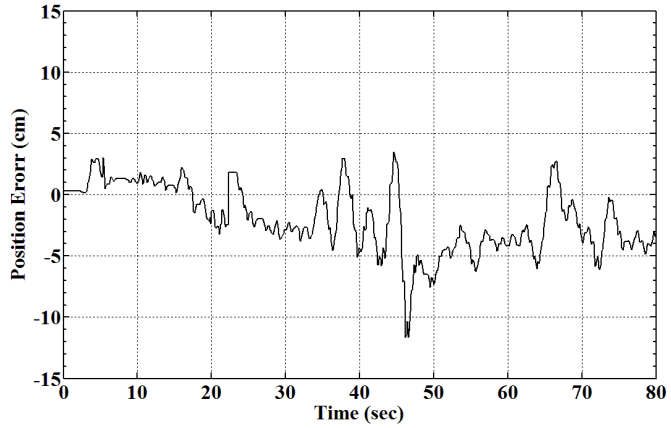


Figure 4.68: ***Position error of the robot with 500g payload moving down***

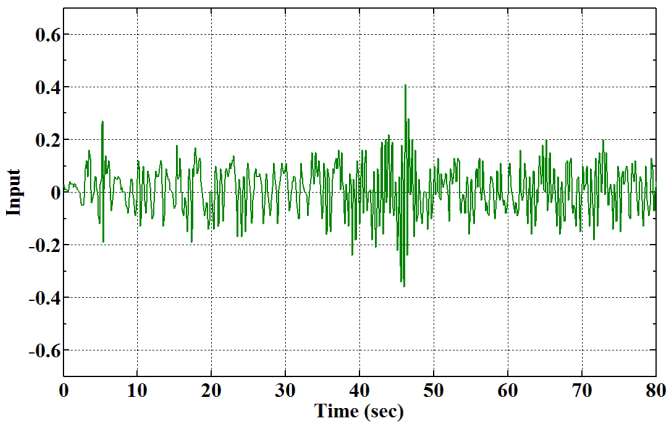


Figure 4.69: ***Normalized input of the robot with 500g payload moving down***

Similar response to the previous tests was observed, however this time the tilt angle measurements became more oscillatory yet acceptable. This oscillation was caused by the increased payload weight which affects the dynamics of the system.

A comparison was made between the two scenarios of the payload motion for both weights. The tilt angle measurements in Figure 4.70 revealed that the stability of the robot was improved by lifting the payload up, in other words moving the global COM higher. Likewise, Figure 4.71 shows that the tilt angle became more oscillatory as the payload moved to a lower position.

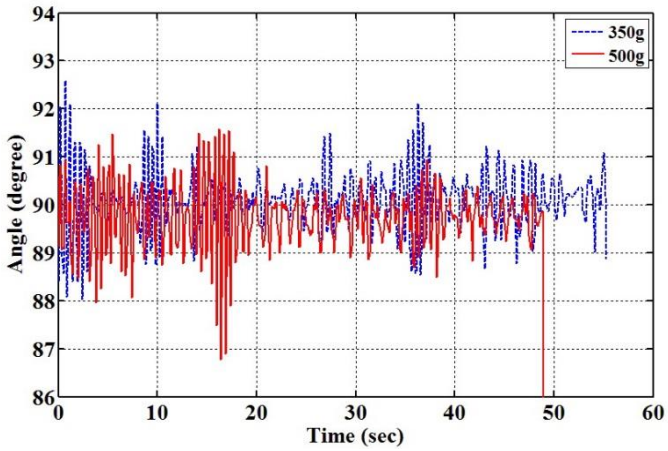


Figure 4.70: ***Tilt angle measurement for moving up payload***

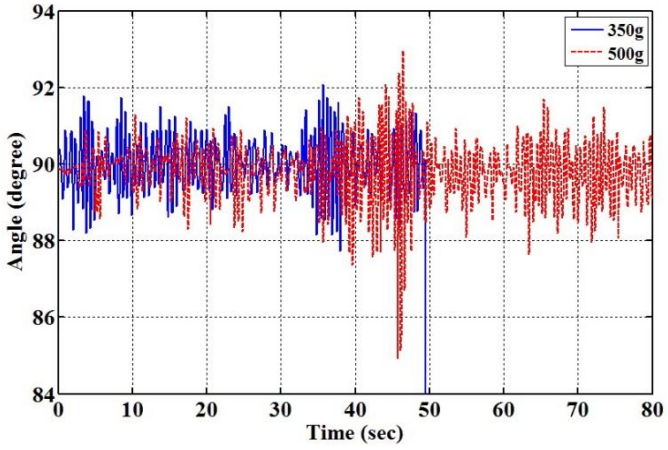


Figure 4.71: ***Tilt angle measurement for moving down payload***

Furthermore, the same relation was noticed between the position of the payload and the normalized input of the motors, the controller effort, as shown in Figures 4.72, and 4.73. This relation leads to a conclusion that the two-wheeled robot would require less energy to maintain its balance for a higher position of COM as explained numerically in Tables 4.4 and 4.5.

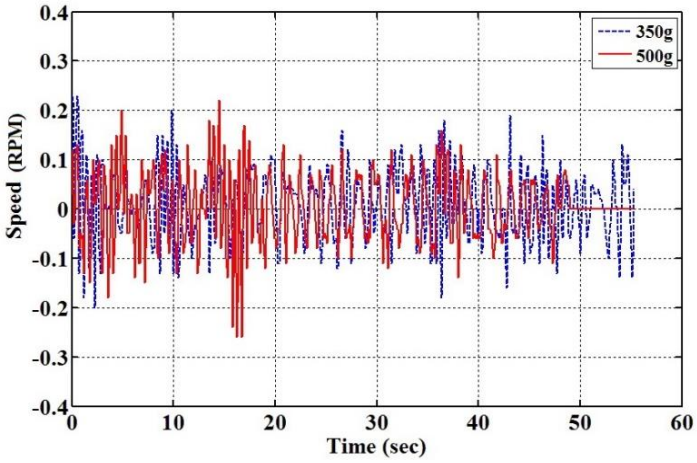


Figure 4.72: ***Normalized speed measurement for moving up payload***

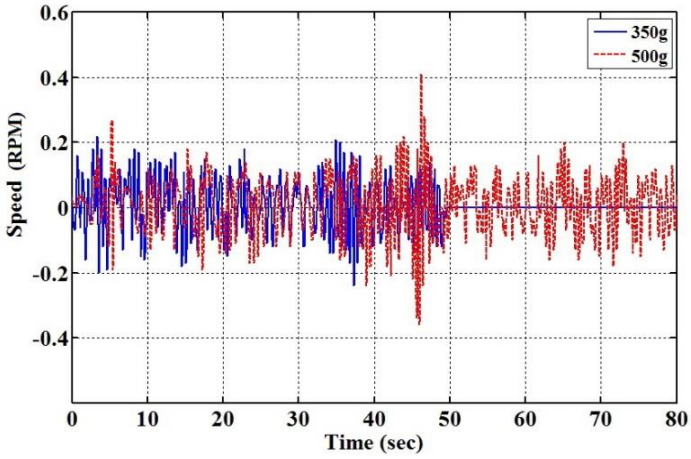


Figure 4.73: ***Normalized speed measurement for moving down payload***

Table 4.4: ***Performance analysis of the two wheeled robot with 350g moving payload***

Payload location		Max Error	Min Error	P-P Error	MAE	RSME	Equivalent energy
Down	Position	4.49	-0.97	5.47	1.12603	1.69948	--
	Angle	2.28	-2.07	4.35	0.67389	0.84086	--
	Input	--	--	--	0.07099	0.08743	5.356
Up	Position	2.17	-2.06	4.23	0.83084	0.97793	--
	Angle	1.81	-1.78	3.59	0.59336	0.74157	--
	Input	--	--	--	0.07602	0.09148	4.7835

Table 4.5: ***Performance analysis of the two wheeled robot with 500g moving payload***

Payload location		Max Error	Min Error	P-P Error	MAE	RSME	Equivalent energy
Down	Position	3.49	-2.41	5.9	1.09383	1.47846	--
	Angle	2.04	-1.48	3.52	0.62039	0.76214	--
	Input	--	--	--	0.06703	0.07988	4.3851
Up	Position	2.09	-2.77	4.86	1.56	1.60221	--
	Angle	1.52	-0.94	2.46	0.37092	0.47857	--
	Input	--	--	--	0.05115	0.06014	3.8687

4.6.3 Inclined surface test

The experiment involved balancing the two-wheeled robot on an inclined plane assuming that the robot was initially balanced and positioned at the middle of the plane, as shown in Figure 4.74.

Figure 4.74: ***Inclined surface experiment setup***

The first experiment was conducted on a 5 degrees inclined surface. Figures 4.75 to 4.77 show the real time performance of the robot in terms of the tilt angle and the normalized wheels speed. As noted the controller has maintained the robot balance effectively keeping the tilt angle within ±1º

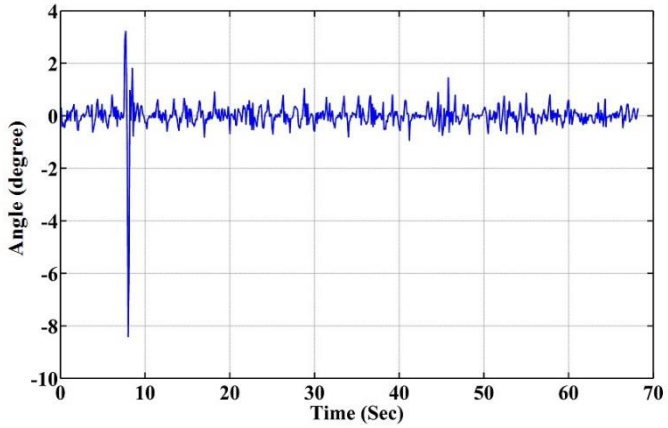


Figure 4.75: ***Tilt angle measurement while balancing on 5° inclined surface***

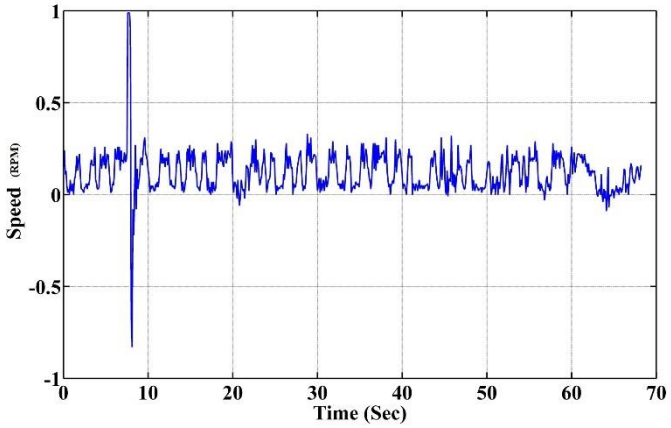


Figure 4.76: ***Normalized wheels speed while balancing on 5° inclined surface***

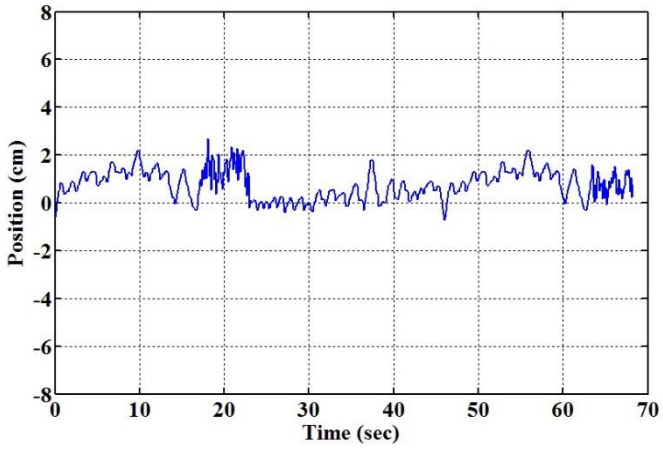


Figure 4.77: ***Position of the robot while balancing on 5° inclined surface***

Two more experiments were conducted on a surfaces with a different inclination angles of 7.5 and 10 degrees. Figures 4.78 to 4.80 illustrate the performance of the robot on 7.5º inclined surface in terms of the tilt angle, the normalized wheels speed and position respectively. Whereas, Figures 4.81 to 4.83 illustrate the performance of the robot on 10º inclined surface.

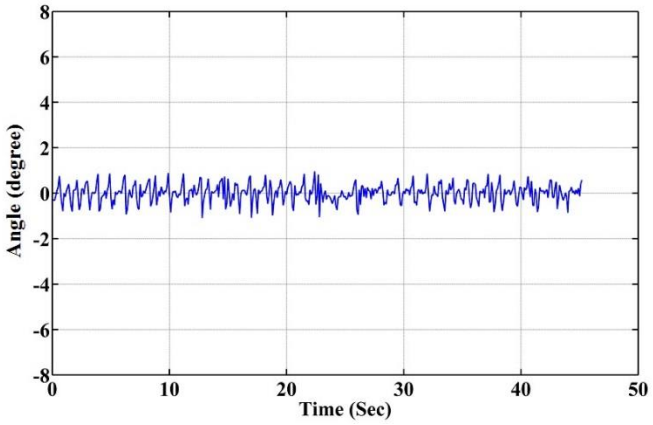


Figure 4.78: ***Tilt angle measurement while balancing on 7.5° inclined surface***

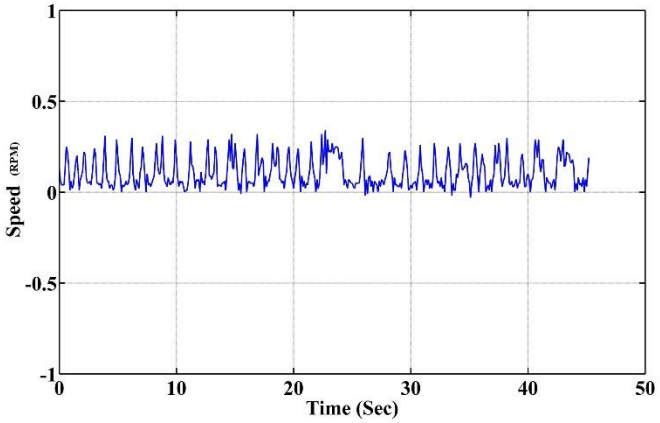


Figure 4.79: ***Normalized wheels speed while balancing on 7.5° inclined surface***

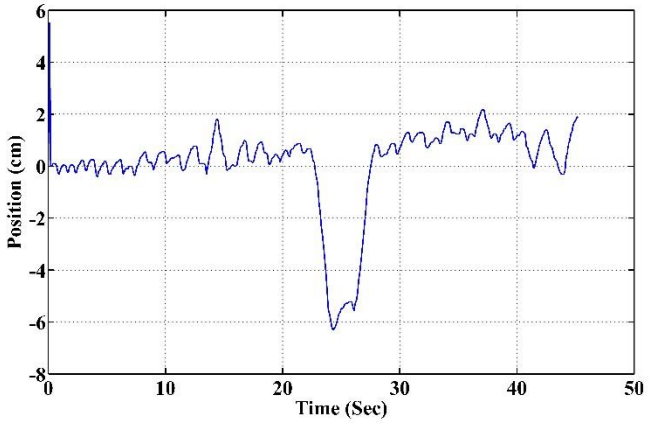


Figure 4.80: ***Position of the robot while balancing on 7.5° inclined surface***

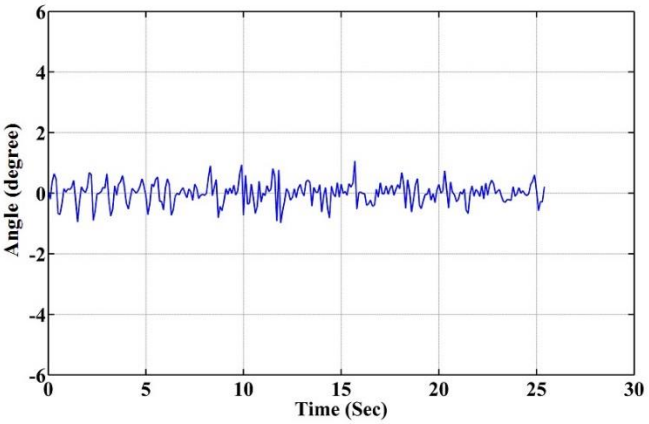


Figure 4.81: ***Tilt angle measurement while balancing on 10° inclined surface***

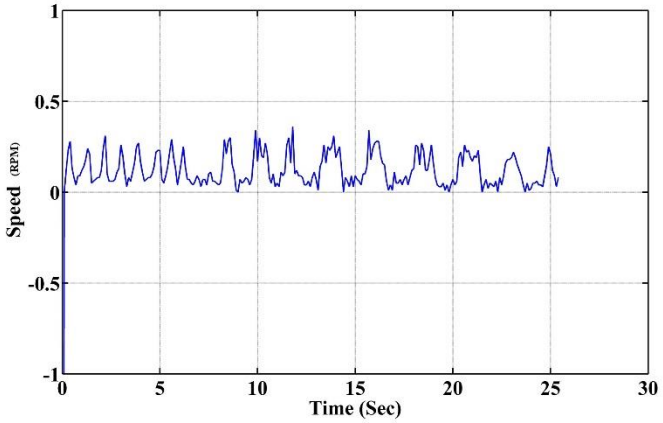


Figure 4.82: ***Normalized wheels speed while balancing on 10° inclined surface***

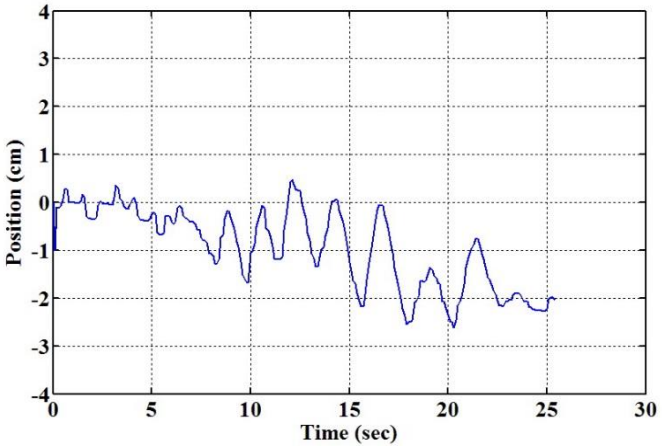


Figure 4.83: ***Position of the robot while balancing on 10° inclined surface***

As noted in Figures 4.75 and 4.78 the controller successfully kept the robot balanced within an acceptable range of the tilt angle error in the two experiments. However it failed to maintain a fixed position of the robot on the 10 degrees inclined surface where the robot started to drag backwards due to the increased horizontal force component of its weight, as depicted in Figure 4.81.

From analysing the results of the normalized wheels speed of the robot, the controller effort, for the three experiments presented in Figures 4.76, 4.79, and 4.82, it is clearly noted that the mean of these measurement data was greater than zero. This was expected as it reflects the required additional wheels torque to counter act the extra drug force due to the weight of the robot.

4.7 Summary

In this chapter the development of a PID based control mechanism has been presented for balancing the two-wheeled robot. The controller has been digitally implemented on the microcontroller development board and tested in real-time. Furthermore, the control structure of the two-wheeled robot has been extended allowing self-erecting of the robot from a resting position. The controller performance was verified by the real-time experiments on the two wheels robot rig.

Further investigation were carried out for the robot response in terms of different payload sizes, dynamic position of the payload, and balancing conditions. The controller of the system was developed to deal with the new design of the robot that provides the mechanism of moving the payload. Experimental studies were carried out to analyse the effect of changing the payload position and size on the two-wheeled robot system. It has been concluded that with the payload positioned higher with more weight the robot becomes more stable and the required controller effort would be less within the system operating range. Further tests were conducted on the two-wheeled robot rig involved balancing the robot on inclined plane with different inclination angles, in order to evaluate the controller performance of the system.

The experimental results have shown the ability of the robot to maintain balance during manoeuvres, and the designed control system has achieved the requirements imposed by low cost, flexibility and stability especially during self-erecting.

Since there are few un-modelled dynamics in the developed model of the two-wheeled robot such as friction and backlash, fuzzy logic control will be considered to control the robot in the next chapter.

CHAPTER 5

Fuzzy logic control of the two wheeled robot

5.1 Introduction

As presented in the previous chapters, the two-wheeled robot system model is non-linear, coupled and under actuated. Also there are other parameters in the system that were neglected in the derivation of the model such as the frictions, gear backlashes, etc. These led to a mathematical model that is not the exact model for the developed two-wheeled robot test rig. Therefore, fuzzy logic control algorithm is considered to develop a controller for the two-wheeled robot.

In this chapter a brief introduction to fuzzy set and fuzzy control theory is presented. Then, the design of a fuzzy controller is presented followed by the real-time implementation of the developed controller on the robot hardware. Finally the experimental results of the system are analysed and discussed.

5.2 Fuzzy logic

The fuzzy set theory concept was suggested by Zadeh in 1965 for the very first time (Zadeh, 1965). According to this theory, humans' reasoning are based on fuzzy sets rather than numbers and discrete symbols. In 1974 Mamdani utilized this theory and put it in practice when he implemented a fuzzy logic controller to a steam engine (Mamdani, 1974). Successively, many practical applications of fuzzy logic were invented, such as the control system of the Subway Sendai Transportation in Japan in 1984, automated aircraft vehicle landing in 1987 and the first fuzzy TV set by Sony in 1990. Nowadays, huge numbers of fuzzy logic intervention and projects are introduced and many control techniques adopt fuzzy logic systems as powerful tools in different areas such as robotics, medicine, instrumentation and manufacturing.

There is a degree of uncertainty or fuzziness in most of the mechanical, electrical, and practical systems because of the lack of exact or perfect mathematical representation of their behaviour. Fuzzy controllers are model-free

where the underlying mechanics are presented linguistically rather than mathematically. Hence, control systems that are based on fuzzy logic are efficient, especially where a system is controlled by an expert operator and is difficult to model, or where there is a common uncertainty or ambiguity.

A typical fuzzy logic controller consists of four main parts; a fuzzifier, a rule-base, an inference procedure, and a defuzzifier. A brief description of the fuzzy sets and fuzzy logic controller is given in the following sections.

5.2.1 Fuzzy sets

A set is a collection of objects of any kind; tables, names, images, books, etc. Usually a set can be determined by naming the members. If B is a set, then the following notation means that u is a member of set B: $u \in B$, and if u is not a member of set B then the notation would be written: $u \notin B$. Any set is said to be a subset of a Universal set. The Universal set contains all possible elements having the nature and the property under consideration (Passino et al., 1998).

A membership function $\mu_A(u)$ quantifies the certainty that the element u belongs to the fuzzy set A as described in Figure 5.1. Each membership function will have a boundary that starts from one point and ends at another. This boundary might be a triangle, a trapezoid or a Gaussian shape, etc. The numbers that are mapped by the membership functions are said to be its members. Figure 5.1 shows an example of a triangular membership function. The membership function of a fuzzy set is a continuous function with range [0,1], and it takes any value in the interval [0,1] (Passino et al., 1998).

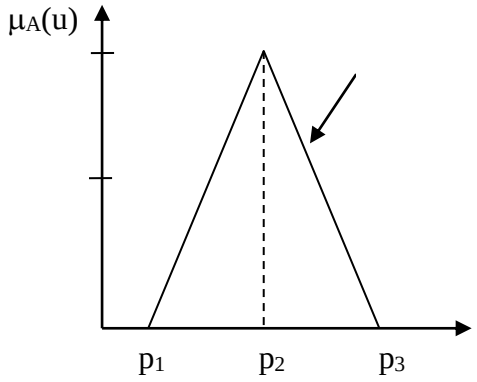


Figure 5.1: ***A triangular membership function***

A fuzzy set in the Universal set U is a set of ordered pairs of an element u and its membership degree $\mu_A(u)$ such as:

$$A = \{(u, \mu_A(u)) / u \in U\} \quad (5.1)$$

5.2.1.1 Operations on fuzzy sets

There are many operations which can be applied on fuzzy sets. Equality, containment, complement, union, and intersection of two fuzzy sets A and B are the basic operations. The fuzzy sets A and B are equal if and only if $\mu_A(u) = \mu_B(u)$ for all $u \in U$ also if $\mu_A(u) \leq \mu_B(u)$ for all $u \in U$ then B contains A ($A \subset B$). The complement of A is a fuzzy set $\bar{A}$ in U which has the following membership function:

$$\mu_{\bar{A}}(u) = 1 - \mu_A(u) \quad (5.2)$$

The union of A and B is a fuzzy set $C = A \cup B$ in U, whose membership function is defined as:

$$\mu_{C=A\cup B}(u) = \begin{cases} \max(\mu_A(u), \mu_B(u)) \\ or \\ \mu_A(u) + \mu_B(u) - \mu_A(u).\mu_B(u) \end{cases} \quad (5.3)$$

The intersection of A and B is a fuzzy set $C = A \cap B$ in U with membership function

$$\mu_{C=A\cap B}(u) = \begin{cases} \min(\mu_A(u), \mu_B(u)) \\ or \\ \mu_A(u).\mu_B(u) \end{cases} \quad (5.4)$$

5.3 Fuzzy controller operation and structure

Fuzzy logic controllers are model-free where the underlying mechanics are presented linguistically rather than mathematically. Fuzzy logic controllers are very functional, especially where a system is difficult to model, is controlled by a human operator or expert, or where ambiguity or vagueness is common. A typical fuzzy system consists of a rule-base, membership functions (MFs), and an inference procedure.

The fuzzy controller structure is divided into three steps:

- *Fuzzification:* the inputs are fuzzified i.e. the crisp inputs are changed to fuzzy inputs.
- *Fuzzy Processing:* using the fuzzy input membership values, the fuzzy rules, and the fuzzy inference, a fuzzy output is obtained.
- *Defuzzification:* Producing a real crisp value from the fuzzy output. This value is a logical control signal that usually goes to the plant.

Figure 5.2 shows the structure of the fuzzy controller with fuzzifier and defuzzifier

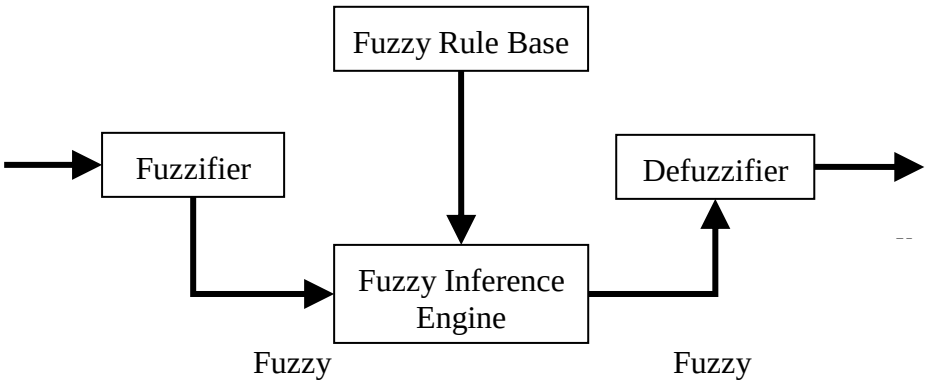


Figure 5.2: ***Basic configuration of fuzzy system***

5.3.1 Fuzzification

Fuzzification is the process of changing a numeric input into a fuzzy input which will have its own membership value (Passino et al., 1998).

A_{u_i} is a fuzzy set (trapezoidal, triangular, Gaussian) MF such that:

$$\mu_{A_{u_i}}(u) = \begin{cases} 1 & if\ u = u_i \\ 0 & decreases\ \ from\ 1\ as\ u\ moves\ \ from\ u_i \end{cases} \tag{5.5}$$

5.3.2 Linguistic variables

Using linguistic variables enables to take words as values of variables (Wang, 1997). Each group of crisp variables is transformed to a fuzzy set which is defined linguistically by naming its membership function, so that human beings can understand these numbers and refer to them in a logical manner. Linguistic variables are the main body behind writing the rules. The linguistic terms are made of the following groups (Passino et al., 1998)

- *The Primary Terms:* these are the basic ones which are used to distinguish between the fuzzy sets, e.g. Small, Medium, or Big.
- *The Connectives:* these connect the input variables by an operation which decides how the fuzzy sets output shall be calculated, e.g. And, Or, Not.
- *The Hedges:* these are added to the primary terms if more fuzzy sets are required to be defined, e.g. Very, Most, Rather, Slightly, More, or Less.

5.3.3 Fuzzy rule base

In fuzzy systems, the human knowledge is represented in terms of IF-THEN rules. The combination of these rules forms the fuzzy rule-base. Fuzzy systems can be divided into two categories depending on the type of their fuzzy rules:

- Standard fuzzy systems: This fuzzy system uses linguistic fuzzy rules (Mamdani-type fuzzy rules) which are formed solely from linguistic variables and values. They are simply abstract ideas about how to achieve good control (Passino et al., 1998). The general form of Mamdani rules is

If premise **Then** consequent

This type of fuzzy systems is used for all the fuzzy logic control strategies in this study.

- Functional fuzzy system: These are known as Takagi-Sugeno-Kang (TSK) fuzzy systems, proposed as an alternative to the standard fuzzy systems (Passino et al., 1998). The TSK rules can be described as follows:

$$IF\ x_1\ is\ C_1^l\ and \cdots and\ x_n\ is\ C_n^l,\ THEN\ \ y^l = c_0^l + c_1^l x_1 + \cdots + c_n^l x_n$$

Where C_i^l are fuzzy sets, c_i^l are constants, and l=1, 2…, M is the rule number. The IF parts of the rules are the same as in the ordinary fuzzy IF-THEN rules, but the THEN parts are linear combinations of the input variables $x = (x_1,...,x_n) \in U$ (Wang, 1997).

Standard fuzzy rules are commonly used. They are easy to understand by a human expert, and the consequent parts are fuzzy. While in functional fuzzy rules the consequent part is a mathematical function. It is more accurate and convenient on mathematical analysis and system analysis (Passino et al., 1998).

5.3.4 Fuzzy inference engine

In a fuzzy inference engine, each rule in the fuzzy rule-base determines an output fuzzy set and the output of the whole fuzzy inference engine is the combination of the individual fuzzy sets selected by the fired rules. The combination can be taken either by union or by intersection (Wang, 1997).

5.3.5 Defuzzification

Defuzzification takes place after the rules have been executed and the fuzzy sets have been combined together according to the predefined operation. Defuzzification is to get the final crisp value from a fuzzy quantity. There are several defuzzification methods such as max-membership method, mean of maxima method and centroid of gravity method. In this study the centroid of gravity (COG) method is used as described in Figure 5.3.

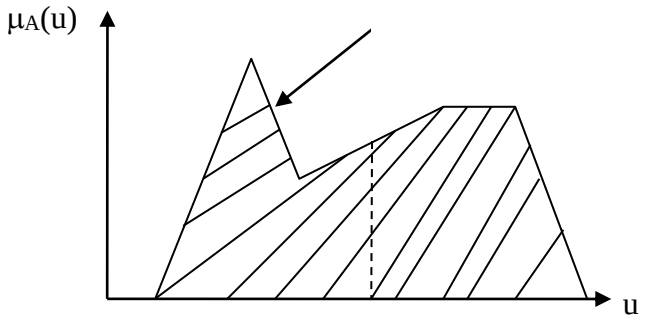


Figure 5.3: ***The COG defuzzification method on a fuzzy output***

u_c is chosen so that it represents the centre of gravity for the shaded area in Figure 5.3, and it can be calculated using:

$$u_c = \begin{cases} \dfrac{\int \mu_A(u).u.du}{\int \mu_A(u).du} & \textit{in the continous case} \\ \dfrac{\sum \mu_A(u).u}{\sum \mu_A(u)} & \textit{in the discrete case} \end{cases} \qquad (5.6)$$

The membership functions must overlap with each other, and the overlap should be at a value more than 0.25 in order to cover any uncertainty of the inputs. The use of narrower membership functions results in a faster response. The choice of defuzzification procedure does not significantly influence a system's performance (Passino et al., 1998).

There are many ways to develop fuzzy rules, such as using manuals, operation instructions and any other available documentation. Another way is by using experienced operators, who understand the process very well. Fuzzy rules can also be developed experimentally by observing the control process, taking important information and building up knowledge of the system. After constructing the rules, the whole set of rules is usually checked to determine if it is complete, consistent, and continuous (Wang, 1997).

Fuzzy controllers differ significantly from conventional controllers. They use the operating experience and physical knowledge in the control process. They do not use mathematical models of the plant, as producing mathematical models can be very time consuming and expensive. Strategies are generated based on a verbal communication, as the rules are written verbally and the fuzzy sets are defined verbally too. Fuzzy logic control is very effective when the system shows uncertainties, non-linarites, and un-modelled dynamics. In this case fuzzy logic control can be said to be able to generate a smooth control, and a robust control behaviour.

5.4 Types of fuzzy controllers

The most three popular types of fuzzy controller structure that have been implemented and investigated in the literature are PD-type, PI-type, and PID-type fuzzy controllers. They provide enhanced performance than their conventional PD, PI, and PID counterparts especially with nonlinear application (Chou and Lu, 1993). On the other hand, PD and PI fuzzy controllers possess the same characteristics as the conventional PD and PI respectively. However, the difficulty with implementing a PID-type fuzzy controller is that it requires three inputs which will increase the number of fuzzy rules leading to more complicated controller design.

5.4.1 PD-type fuzzy controller

Practically, PD-type fuzzy controllers are more common than other types such as PI-type, due to their good performance in the transient sate. However, PD –type controllers do not have the ability to eliminate the steady-state error. The controller has two inputs, the system error (e) and the change of error (de) to generate the control signal as

$$u(t) = K_p\, e(t) + K_d \frac{de(t)}{dt} \tag{5.7}$$

Figure 5.4 depicts the general block diagram of a PD-type fuzzy controller. The linguistic variables of the membership functions for the inputs and the outputs are named as in Table 5.1, assuming that there are five MFs for the inputs and output of the controller. The fuzzy rule base of the PD-type controller is listed in Table 5.2 (Passino et al., 1998).

Table 5.1: ***Linguistic hedges of the MFs***

Label	Hedge
NB	Negative big
NS	Negative small
Z	Zero
PS	Positive small
PB	Positive big

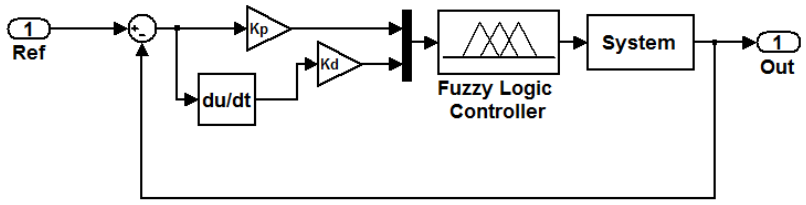


Figure 5.4: ***PD-type fuzzy controller structure***

Table 5.2: ***PD-Type fuzzy rules set***

e \ Δe	NB	NS	Z	PS	PB
NB	NB	NB	NB	NS	Z
NS	NB	NB	NS	Z	PS
Z	NB	NS	Z	PS	PB
PS	NS	Z	PS	PB	PB
PB	Z	PS	PB	PB	PB

5.4.2 PI-type fuzzy controller

It is well known that PI controller offers very good performance in eliminating the steady-state error. Similarly, PI-type fuzzy controller reduces the steady-state error of the system. There are two structures of the PI-type fuzzy controller based on the output calculation. The first structure, shown in Figure 5.5, is called absolute PI-fuzzy controller. It has two inputs, the system error (e) and the sum of error (∑e) to generate the control signal as

$$u(t) = K_p e(t) + K_i \int e(t) \tag{5.8}$$

Table 5.3 lists the fuzzy rule base of the absolute PI-type controller assuming that there are five MFs for the inputs and output of the controller (Passino et al., 1998).

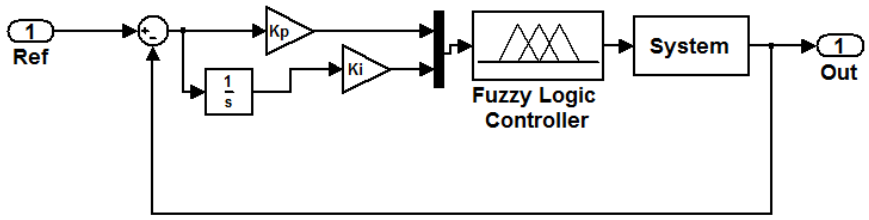


Figure 5.5: ***PI-type fuzzy controller structure***

Table 5.3: ***Absolut PI-type fuzzy rules set***

e \ ∑e	NB	NS	Z	PS	PB
NB	PB	PB	PB	PS	Z
NS	PB	PS	Z	Z	NS
Z	PS	Z	Z	Z	NS
PS	PS	Z	Z	NS	NB
PB	Z	NS	NB	NB	NB

The second structure of the PI-type controller, shown in Figure 5.6, is referred to incremental PI-type controller. Basically, it is a PD-type fuzzy controller with an integrator located at the controller output after the defuzzification block as described below by rewriting the absolute PI-type fuzzy controller Equation (5.8) into the following:

$$\frac{du}{dt} = K_p \frac{de(t)}{dt} + K_i\, e(t) \tag{5.9}$$

$$u(t) = \int \left[K_p \frac{de(t)}{dt} + K_i\, e(t) \right] \tag{5.10}$$

Equation (5.10) is very similar to the PD-type fuzzy controller, however it gives the change in the output rather than the output. Therefore, integrating the output of the fuzzy PD-type controller will generate a control action equivalent to fuzzy PI-type controller. In this configuration the control action is calculated based on the error and the change of error of the system with the PD-type fuzzy rules.

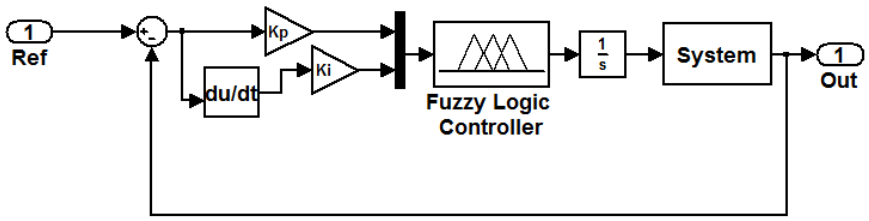


Figure 5.6: ***Incremental PI-type fuzzy controller structure***

5.4.3 PID-type fuzzy controller

A full PID-type fuzzy controller can be implemented in order to meet the system requirements such as fast rise time, minimum overshoot, and shorter steeling time with zero steady-state error as shown in Figure 5.7. However, PID-type fuzzy controller implies three input fuzzy system which will increase the rule base size and make the controller design more complicated. For instance, a fuzzy system with three inputs and one output and five MFs per input and output requires 125

fuzzy rules to build the complete rule base. Therefore, PID-type fuzzy logic controllers are less common and rarely used.

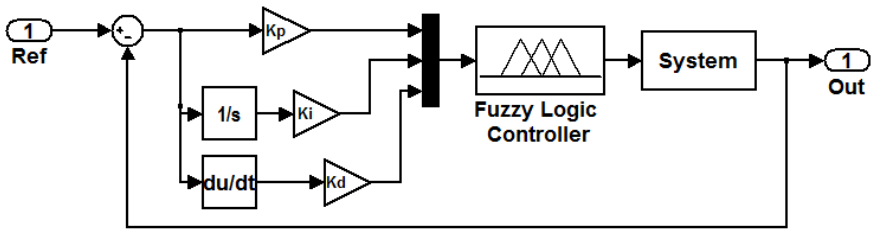


Figure 5.7: ***PID-type fuzzy controller structure***

5.5 Design of the fuzzy logic controller

The essential aim in this project is to maintain the upright balance of the developed two-wheeled robot. Since two-wheeled robot systems are non-linear and there are un-modelled dynamics in their mathematical representation, fuzzy logic controller would be the appropriate choice to put the system in balance state by applying a suitable control signal. Choosing the inputs of the controller is an exceptionally basic procedure, as it is critical that all the data required about the plant is accessible through the controller inputs, to guide the system and achieve high-performance. The inputs of the fuzzy controller developed here are chosen to be the same as the inputs of the controller in the previous chapter which are the error in tilt angle of chassis, and the change of error. The change of error is the same as the rate of change of the tilt angle or the gyro sensor signal. The output of the fuzzy controller is fed to the motors driver to produce the required torque and keep the two-wheeled robot balanced. Figure 5.8 depicts the fuzzy logic controller block diagram of the two-wheeled robot.

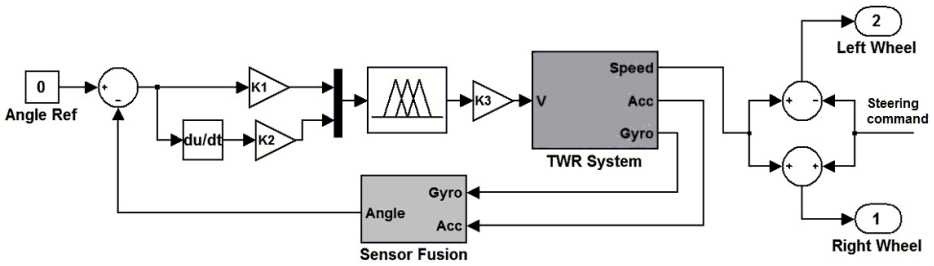


Figure 5.8: ***Two-wheeled robot fuzzy logic controller structure***

The next stage of the design process is the fuzzification procedure, where the real values of the inputs will be translated to linguistic variables. These variables describe the time varying fuzzy controller inputs and outputs following the meaning of the linguistic values that are quantified by a membership functions. Generally, the number of the linguistic variables, membership functions, can be any number, however the common number of the membership functions in the literature is five. Hence, five MFs are chosen for the inputs and the outputs of the controller. The distribution of the MFs is chosen to be equal with 50% overlap that gives sufficient satisfaction for the control process as shown in Figures 5.9 and 5.10. These assumption will simplify the tuning process of the controller and reduce the tuneable parameters to three scaling gains (K_1, K_2, and K_3) of the inputs and the output.

Although Gaussian shape for the MFs is recommended, the triangular shape for the MFs is used in order to speed up the real-time calculation in the hardware. Also the smoothness of the output is not that important for the two-wheeled robot system, as the motors driver reacts as low pass filter on the control signal. Therefore, the performance of the fuzzy controller is not affected so much when using either type of the MFs.

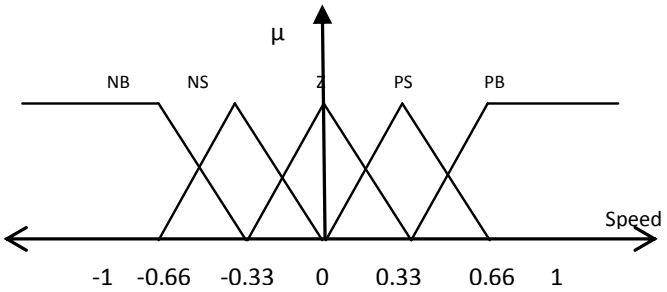


Figure 5.9: ***Output membership functions***

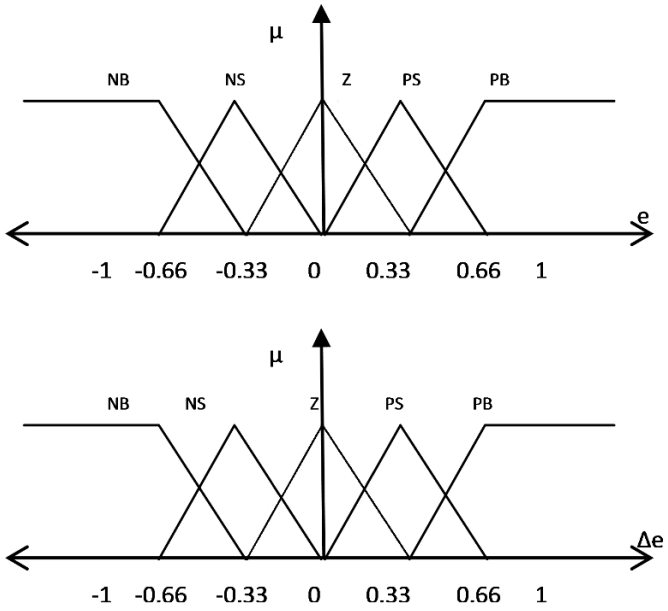


Figure 5.10: ***Inputs membership functions***

The next step is to build the rule base of the fuzzy controller. The maximum number of the rules is determined by the number of inputs and the number of MFs for each input. Since the fuzzy controller has two inputs with five MFs each, the fuzzy rules table will consist of 25 rules. The rules can be determined using the experience of the designer, due to the well understanding of the control process. They also can be tuned by observing the control process, taking important information and building up knowledge on the system. The fuzzy rules for this system are listed in Table 5.4, as the controller represents a PD-type fuzzy logic controller. The rules base for this controller are complete, where any combination

of two linguistic variables fires at least one rule. They are consistent, with no contradictions, and are continuous as shown in Figure 5.11.

Table 5.4: ***PD-type fuzzy controller rules set***

e \ Δe	NB	NS	Z	PS	PB
NB	NB	NB	NB	NS	Z
NS	NB	NB	NS	Z	PS
Z	NB	NS	Z	PS	PB
PS	NS	Z	PS	PB	PB
PB	Z	PS	PB	PB	PB

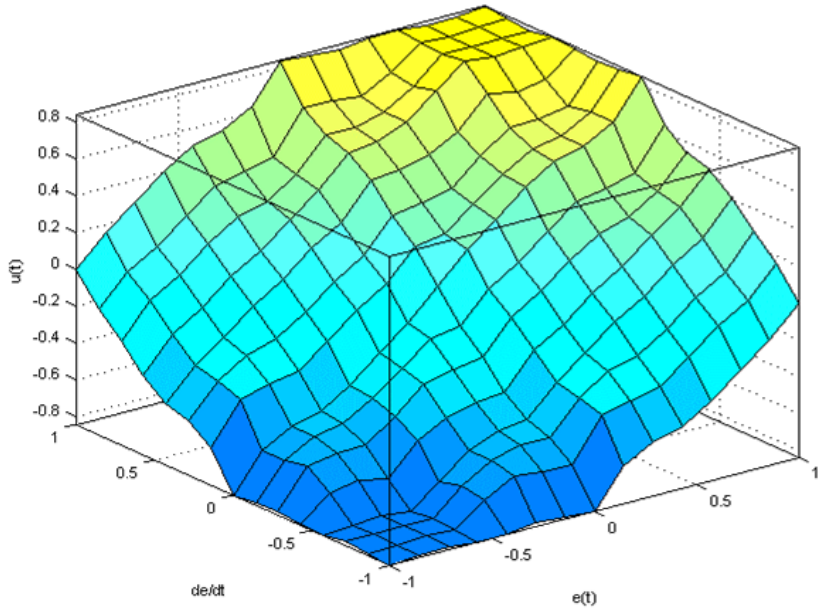


Figure 5.11: ***PD-type fuzzy rules surface***

The final step of the design process is to choose a defuzzification method which converts the fuzzy output of the controller into crisp values. The centroid of gravity method was chosen for this controller as it is the commonly used defuzzification method in feedback control due to its smooth output.

Choosing the right scaling gain values K_1, K_2, and K_3 of the fuzzy controller was achieved through the interface PC software, which provides the user with the ability to monitor the internal parameters of the system and adjust the gains accordingly. Also it provides a wireless interface with the robot where navigation commands can be sent.

5.6 Implementation of fuzzy control

Fuzzy logic controllers can be implemented in real-time on hardware such as FPGA and DSP as input / output lookup table. Such implementation has a limitation of input /output range resolution, as the higher resolution the bigger the table and memory size. Here a new technique is developed to eliminate the limitation of lookup table implementation of the fuzzy logic controllers and to save the memory space. This implies there is an instantaneous output value calculated for every new value of the inputs. This technique is developed for implementation on microcontrollers including 8-bit microcontrollers such as the AVR. It can be adopted to implement any Mamdani type fuzzy controllers with up to 8 inputs and 8 outputs with up to 8 MFs for every input and output of the controller. The user can easily design a fuzzy controller and load it into the microcontroller. The aim of the following sections is to provide a detailed description of the implementation of the fuzzy logic controller parts.

5.6.1 Fuzzy logic controller coding

In order to implement a fuzzy controller digitally in real-time, all of its parameters have to be coded into binary form. This can be done using the developed fuzzy software which was described in chapter 2. The fuzzy software generates a binary file for the coded parameters of the fuzzy controller such as number of inputs and output, number of the MFs for each input/output, the types of MFs, and the rules base of the controller. Since the technique is developed to be compatible with 8-bit microcontrollers, the maximum storage size of the parameter is assumed to be 8 bits or one byte.

5.6.1.1 Input/output MF coding

At the beginning, the number of inputs of the fuzzy controller is defined by a single byte of data followed by eight bytes that define the number of MFs of each input. This structure is used to define the number of outputs of the fuzzy controller and their MFs as well. Then, each membership function / linguistic variable of the inputs and outputs in the fuzzy controller is defined by means of point P1, P2, P3 and P4, where each point is assigned one byte. Therefore a trapezoidal membership

function can be defined by using the straight line equation as shown in Figure 5.12. The value of the MF $\mu_A(x)$ or the certainty that x belongs to the linguistic variable μ_A can be calculated as

$$\mu_A(x) = \begin{cases} \dfrac{x - P1}{P2 - P1} & if \;\; P1 < x < P2 \\ 1 & if \;\; P2 \leq x \leq P3 \\ \dfrac{P1 - x}{P4 - P3} & if \;\; P3 < x < P4 \\ 0 & otherwise \end{cases} \quad (5.11)$$

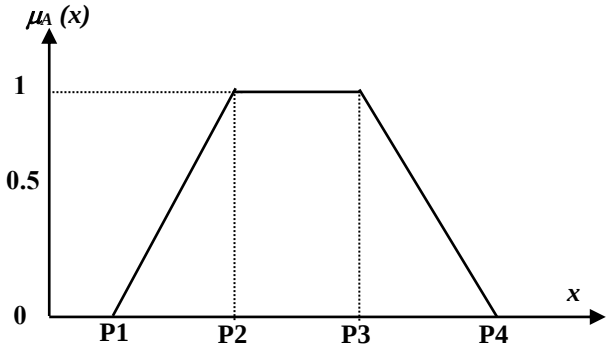
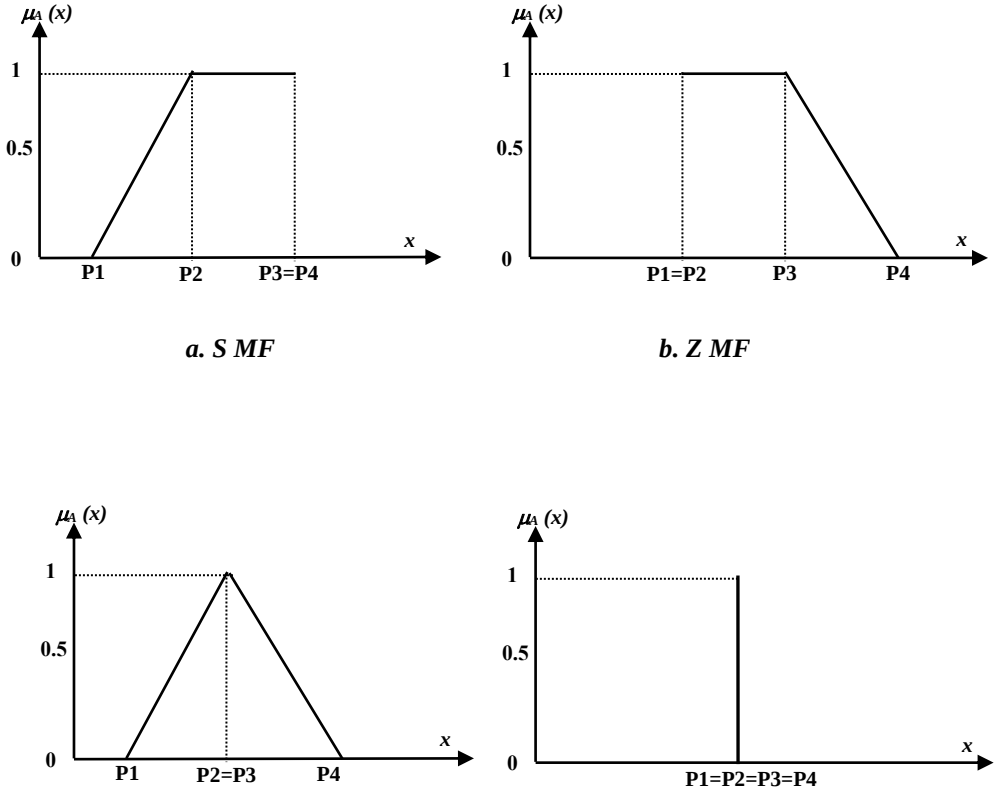


Figure 5.12: ***Linguistic variable representation***

Other types of membership function can be derived from the trapezoidal MSF using the same technique with small change to the assigned values of the four bytes. For instance, when the first byte is assigned a value equal to the second byte, i.e. P1 = P2, then (P2, P2, P3, P4) represents a left rectangle membership function and (P1, P2, P3, P3) will represent a right rectangle membership function; As the segment between P1 and P2 or P3 and P4 will be a dot rather than a line. Similarly, a triangle MF is represented by (P1, P2, P2, P4) or (P1, P3, P3, P4), and a singleton MF by (P1, P1, P1, P1) as described in Figure 5.13.

The coded bytes of the number of inputs\outputs, number of MFs, and the points of each MF are stored in a binary file or memory at predetermined sequence and location followed by the data of the coded fuzzy rules base as shown in Figure 5.14.



a. S MF *b. Z MF*

c. Triangle MF *b. Singleton MF*

Figure 5.13: ***Different membership function representation***

No. of Inputs	1000h
No. of MF for Input0	1001h
No. of MF for Input1	1002h
* *	
No. of MF for Input7	1008h
B1 of MF0 for Input0	1009h
B2 of MF0 for Input0	100Ah
B3 of MF0 for Input0	100Bh
B4 of MF0 for Input0	100Ch
B1 of MF1 for Input0	100Dh
* *	
B1 of MF7 for Input7	1105h
B2 of MF7 for Input7	1106h
B3 of MF7 for Input7	1107h
B4 of MF7 for Input7	1108h
No. of Outputs	1109h
No. of MF for Output0	110Ah
No. of MF for Output1	110Bh
No. of MF for Output2	110Ch
B1 of MF0 for Output0	110Dh
B2 of MF0 for Output0	110Eh
B3 of MF0 for Output0	110Fh
B4 of MF0 for Output0	1110h
* *	
B1 of MF0 for Output2	1169h
B2 of MF0 for Output2	116Ah
B3 of MF0 for Output2	116Bh
B4 of MF0 for Output2	116Ch
Fuzzy Rule Table Storage Area	116Dh
	1FFFh

Figure 5.14: ***Fuzzy parameters binary file/memory example***

5.6.1.2 Fuzzy rule base coding

The fuzzy rules base of the fuzzy controller is coded into binary and stored in the parameter binary file. Every rule is converted into few bytes depending on the number of consequents and antecedents. For instance, a fuzzy rule which has a single consequent (output) and two antecedents (input) will be three bytes long, as every consequent or antecedent in the fuzzy rule is coded by a single byte.

First, assign a sequence number to every MF starting with 0 for the most left hand MF, then the rule bytes are coded as bellow (shown in Figure 5.15)

- The first three bits (D_2 D_1 D_0) represent the MF sequence number associated with the input/output in the rule part, hence (000) corresponds to the first MF and (111) for the last MF of the input/output.
- The next three bits (D_3 D_4 D_5) define the input/output number in the rule part; hence (000) corresponds to input/output number 1 and (111) for number 8.
- The next bit (D_6) defines the fuzzy operator between the rule parts, 0 for AND operator, and 1 for OR. Also it signals the end of the rules table when both D_6 and D_7 are set to 1.
- The last bit (D_7) determines if the byte represents a consequent part or antecedent part of the fuzzy rule.

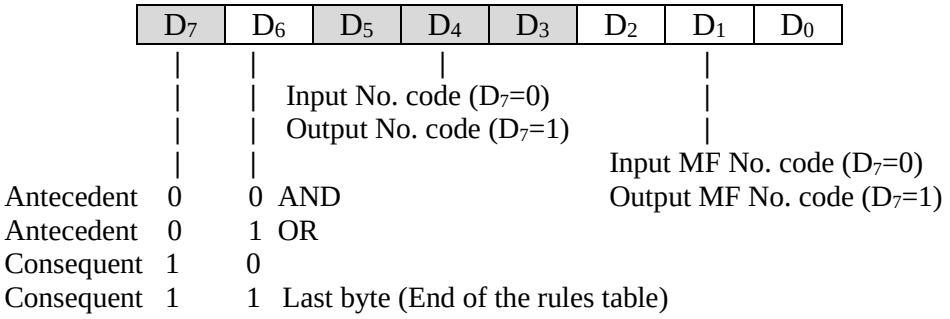


Figure 5.15: ***Fuzzy rule byte coding***

Based on this assumption the biggest fuzzy controller that can be coded using this technique will have 8 inputs, 8 outputs, and 8 MFs for every input/output. The longest theoretical fuzzy rule will be 16 bytes long assuming that it tests all the possible inputs and updates all the possible outputs. In this study, there are 25 fuzzy rules of three bytes per rule as the system has two inputs and a single output with five MFs each. Therefore, the length of the coded fuzzy rules is 75 bytes only.

5.6.1.3 Coding example

Typical fuzzy logic system of three inputs and two output is considered as example for the coding process. Every input and output of the system assigned with five MFs as illustrated in Figure 5.16. The rules base of the system has two fuzzy rules as listed below:

If In_1 is NS AND In_2 is PS then Out_0 is Z

If In_1 is Z OR In_3 is S then Out_0 is B AND Out_1 is PB

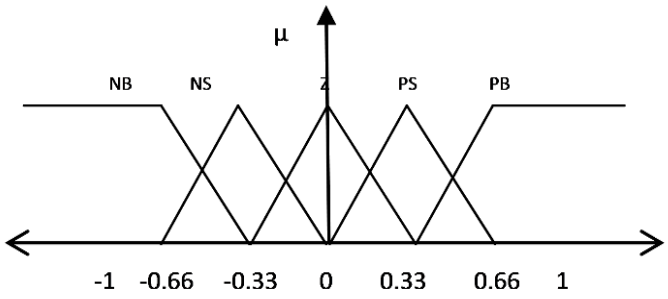


Figure 5.16: ***Input / Output membership functions***

The coding will start with giving a sequence code for every MF of the inputs and outputs starting from the left hand MF as listed in Table 5.5.

Table 5.5: ***The sequence code of the input/output MF***

Label	Code
NB	**000**
NS	**001**
Z	**010**
PS	**011**
PB	**100**

Then each part of every rule in the fuzzy rules table is coded into single binary byte according to the pattern in Figure 5.15. Hence, the first rule will be three bytes long and four bytes for the second rule as described in Figures 5.17 and 5.18. Since the second rule is the last one in the fuzzy rules table, the D_6 in the fourth byte is set to (1) in order to indicate the end of the coded rules table.

The fuzzy data PC software, described in chapter 2, was used to perform the coding process of the fuzzy parameters and rule base table. Then it generates two binary files for the parameters of the input/output MFs and the coded fuzzy rules.

If In_1 is NB **AND** In_2 is PB **then** Out_1 is Z

1st byte:				In_1	is	NB		
	0	X	0	0	1	0	0	0
2nd byte:				In_2	is	PB		
	0	0	0	1	0	1	0	0
3rd byte:				Out_1	is	Z		
	1	0	0	0	1	0	1	0

Figure 5.17: ***Coding example of the first rule***

If In_2 is Z **OR** In_3 is PS **then** Out_0 is NS **AND** Out_1 is PB

1st byte:				In_2	is	Z		
	0	X	0	1	0	0	1	0
2nd byte:				In_3	is	PS		
	0	1	0	1	1	0	1	1
3rd byte:				Out_0	is	NS		
	1	0	0	0	0	0	0	1
4th byte:				Out_1	is	PB		
	1	1	0	0	1	1	0	0

Figure 5.18: ***Coding example of the second rule***

5.6.2 Fuzzy controller operation and rules decoding

All fuzzy operations and calculations were programmed in C language and implemented on the microcontroller development board. These operations include input fuzzification, fuzzy rules firing, and fuzzy output defuzzification. The parameters of the fuzzy controller were stored in the microcontroller memory in a predefined locations as they were coded into a binary file.

After taking a new sensor measurement, the program in the microcontroller starts fuzzifying the inputs based on the stored MF parameters. Then the fuzzy rules are tested one by one till the end of the rules table, where the end of the rules table was marked by a single byte which contains 0xC0 value. After that, the fired

rules is decoded in order to generate the fuzzy output. Finally, the crisp controller output is calculated using the centroid defuzzification method. This cycle is repeated for every new sample of the inputs. The flow chart of the microcontroller fuzzy program is shown in Figure 5.19.

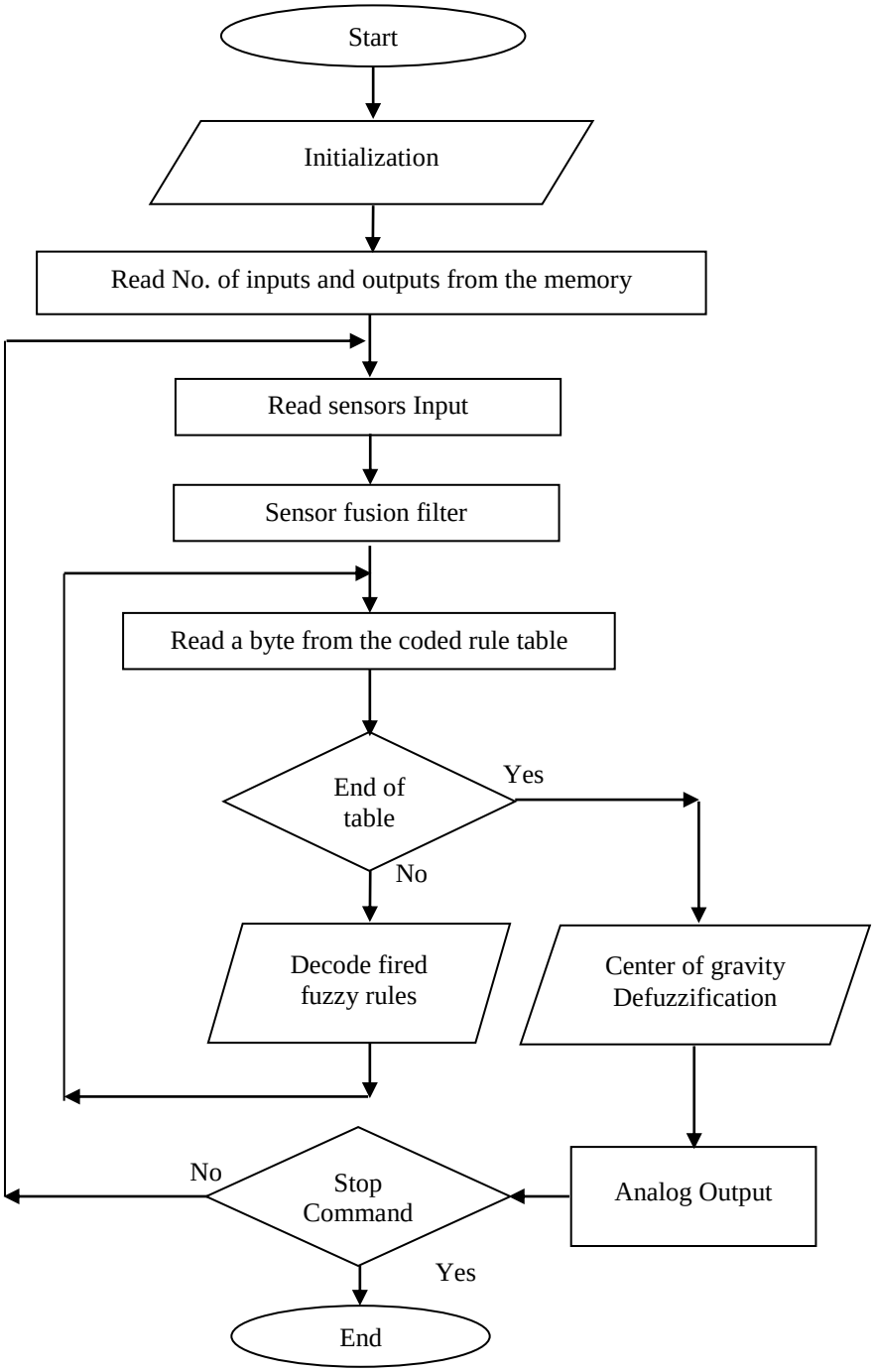


Figure 5.19: ***The flow chart of the Fuzzy controller program***

5.7 Experimental results

In order to evaluate the implementation of the fuzzy controller, a balance test was conducted on the two-wheeled robot rig. The gains of the controller were tuned heuristically to the best possible values using the PC monitor software. Then the experimental results were logged and plotted in Figure 5.20 for the tilt angle and Figure 5.21 for the normalized speed of the wheels. It is clearly noted that the controller managed to stabilize the robot at an upright position with small angle error of ±0.75^{o}.

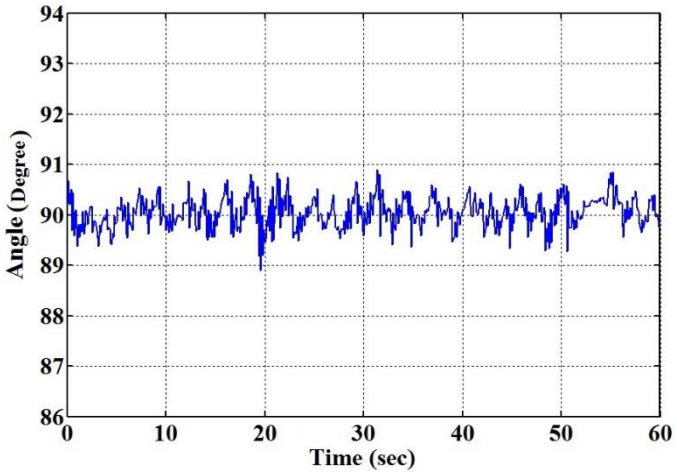


Figure 5.20: ***Experimental results of the tilt angle with fuzzy controller***

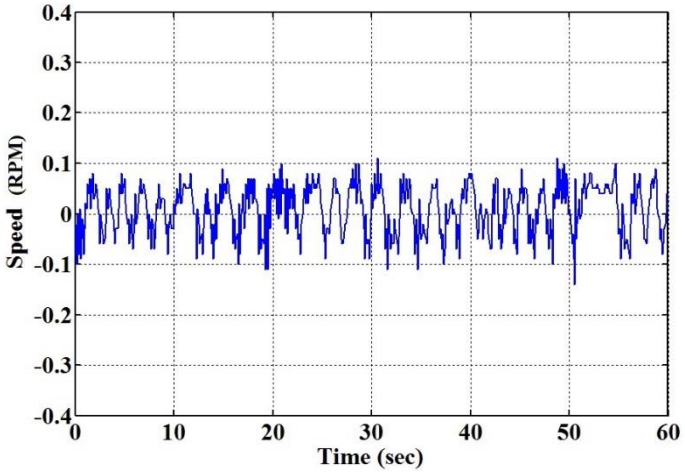


Figure 5.21: ***Experimental results of the normalized speed with fuzzy controller***

5.7.1 Fuzzy controller vs PID

The performance of the two-wheeled robot with the developed fuzzy controller was compared with the performance of the PID based controller which was developed earlier in chapter four. Both of the controllers were tuned to the best possible response heuristically. Figures 5.22, and 5.23 reveal that the fuzzy controller offered a better response than the PID, as the oscillations in the tilt angle became small. Similar effect was noticed in the normalized speed of the motors which implies that less energy is required to balance and stabilize the two-wheeled robot with fuzzy controller. Table 5.6 shows a numerical comparison for both controllers' performance.

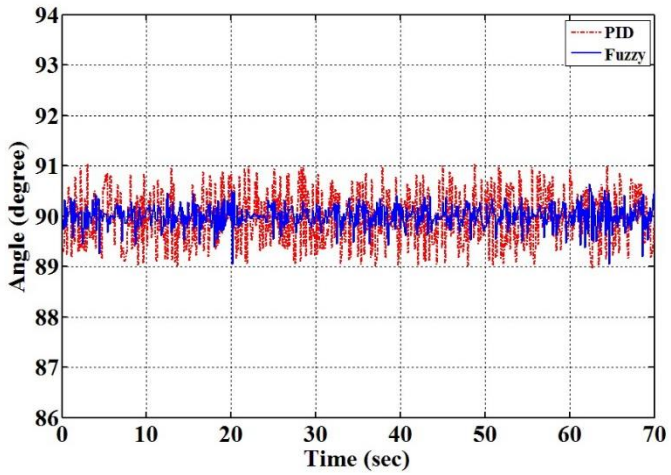


Figure 5.22: ***Experimental results of the fuzzy controller vs PID***

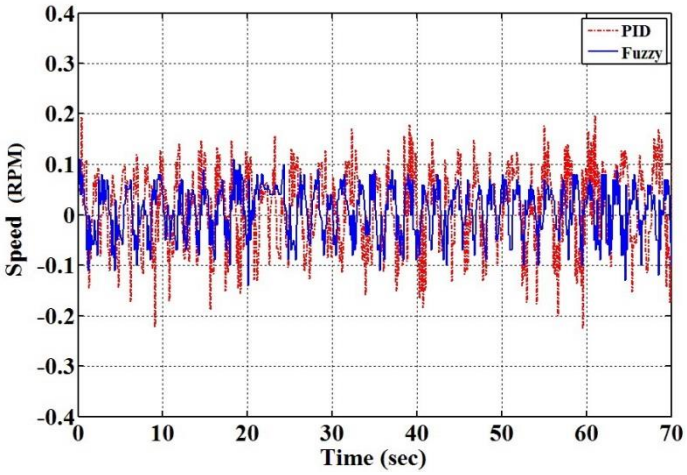


Figure 5.23: ***Experimental results of the fuzzy controller vs PID***

Table 5.6: ***Fuzzy vs PID controller numerical comparison***

		MAE	RMSE	Max	Min	P-P Error	Equivalent energy
PID	Angle	0.4178	0.255	1.0576	-1.011	2.0686	
	Speed	0.0652	0.0064	0.2375	-0.2016	0.4392	4.5893
Fuzzy	Angle	0.1581	0.0394	0.9386	-0.618	1.5567	
	Speed	0.0427	0.0025	0.1362	-0.1155	0.2516	3.0071

5.7.2 Disturbance test

A disturbance test was conducted in order to evaluate the robustness and stability of the developed fuzzy controller. The test was performed by applying three external disturbance pulses to the two-wheeled robot. These pulses were applied horizontally to the chassis of the robot at different times 17, 28 and 36 seconds. The maximum deviation in the tilt angle due to these disturbances was about +4 degrees. The controller responded to the disturbances and stabilized the two-wheeled robot within relatively short time. The experimental results in Figures 5.24 and 5.25 show the performance of the controller for both tilt angle of the chassis and speed of the wheels. It is noted that the controller successfully kept the robot balanced in the presence of the external disturbance.

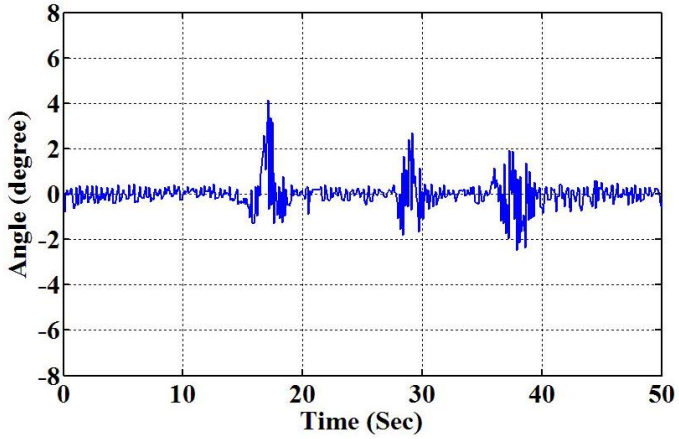


Figure 5.24: ***Tilt angle experimental result of the fuzzy controller with disturbance***

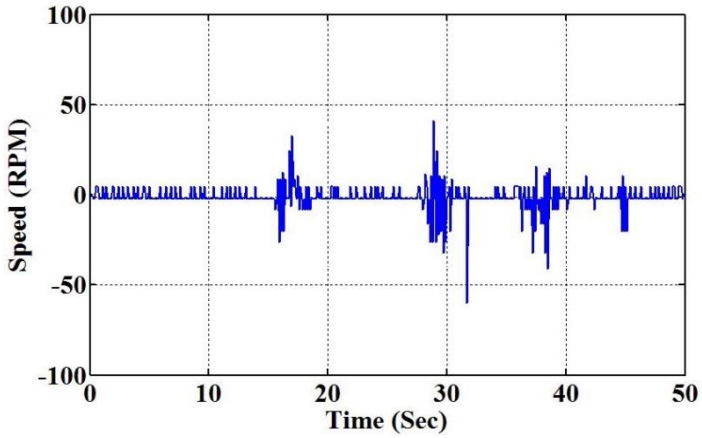


Figure 5.25: ***Wheels speed experimental result of the fuzzy controller with disturbance***

5.7.3 Extra payload weight

Adding extra payload weight to the two-wheeled robot will affect the dynamics of the system. Assuming that the extra weight is added on the top layer of the two-wheeled robot, there are two possible scenarios. First, the weight is placed at the centre of the top layer parallel to the wheels axis and aligned with original centre of mass (COM) of the robot. In such case, the impact of the extra weight on the two-wheeled robot balance and position stability is little as the COM of the robot will be shifted upward along the z-axis.

On the other hand, adding an extra weight at either side of the top layer of the robot, scenario two, will have a considerable impact on the dynamics of the system as the COM of the robot will be shifted towards the extra weight side. Naturally, the two-wheeled robot will drift away forward/backward, in the direction of the extra load, as the COM is shifted from the vertical axis of the robot. The controller of the robot should maintain the balance of the robot. Therefore, the error of the position controller, outer loop, will increase due to the movement of the robot and accordingly will adjust the set point of the balancing controller, inner loop. The final controller action will result in balancing the two-wheeled robot but at an angle other than 0 degree.

A practical test was conducted to evaluate the performance of the controller when an extra weight of about 150g is added on either side of the top layer of the

two-wheeled robot. Initially, the extra load was placed on the frontal side of the top layer of the two-wheeled robot then removed after few seconds. As a result the two-wheeled robot was drifted forward for about one metre in the direction of the added weight then balanced at the new position. Similar inverse action was observed when the extra load was removed, as the two-wheeled robot drifted backwards to its initial position whilst maintaining balance as shown in Figure 5.26. The controller of the two-wheeled robot successfully maintained the balance state during the experiment and kept the tilt angle close to the angle set point as depicted in Figures 5.27 and 5.28.

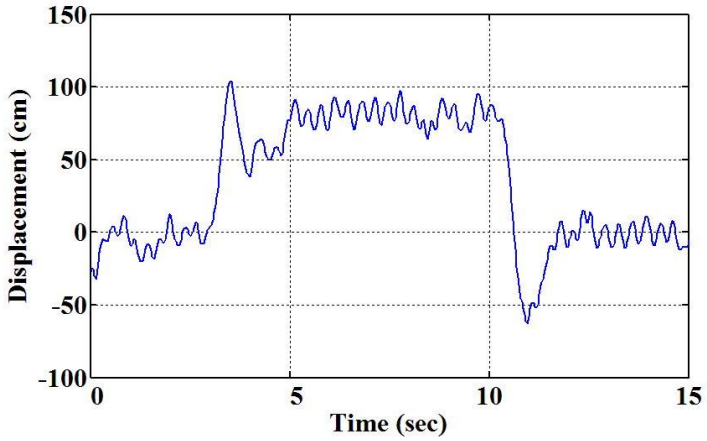


Figure 5.26: ***Displacement of the robot while adding and removing the extra load at the front of the top layer of the robot***

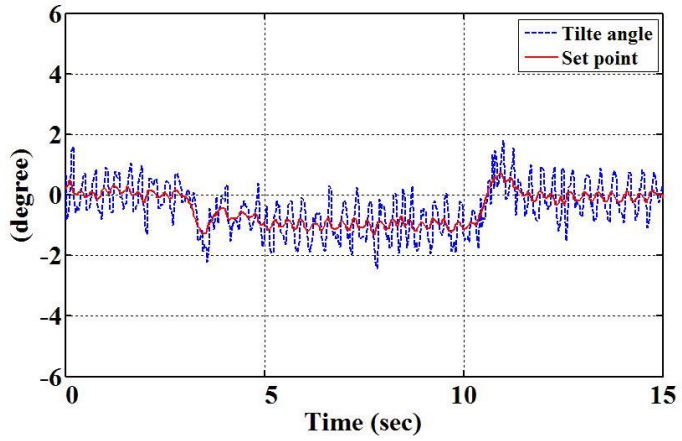


Figure 5.27 ***Tilt angle while adding and removing the extra load at the front of the top layer of the robot***

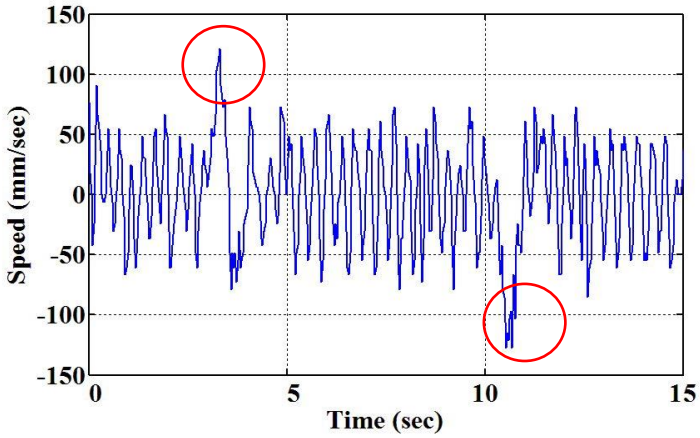


Figure 5.28: ***Travel speed while adding and removing the extra load at the front of the top layer of the robot***

The extra payload weight experiment was repeated but with placing the extra load on the other side, the back side, of the top layer of the two-wheeled robot. As expected, the robot drifted about one metre from its original position in the direction of the extra weight. Then the two-wheeled robot moved back to its original position when the extra load was removed whilst maintaining the balance state of the robot. Figures 5.29 to 5.31 illustrate the performance of the two-wheeled robot when an extra weight is added to and removed from the back side of the top layer of the robot chassis.

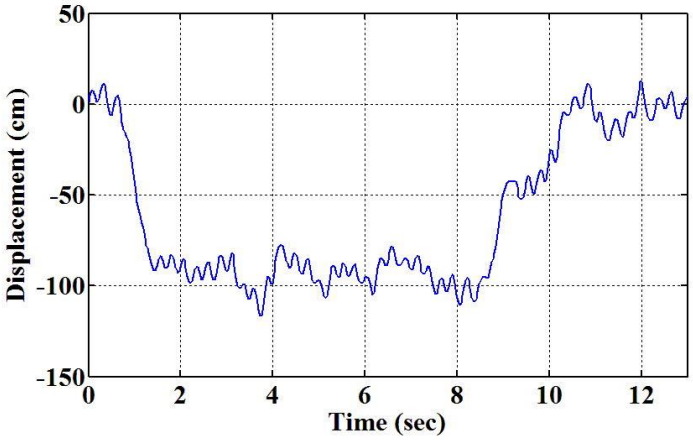


Figure 5.29: ***Displacement of the robot while adding and removing the extra load at the back of the top layer of the robot***

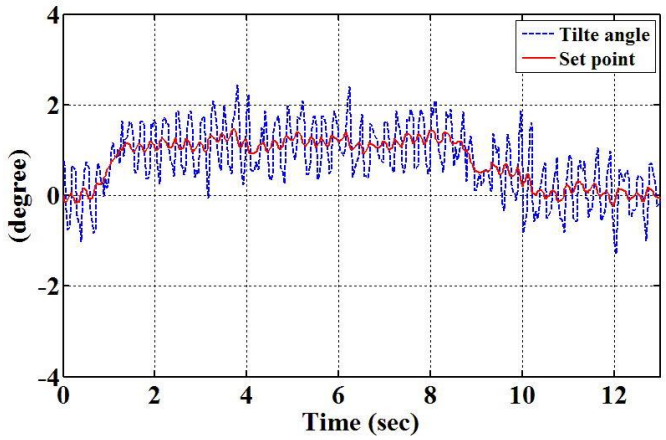


Figure 5.30 ***Tilt angle while adding and removing the extra load at the back of the top layer of the robot***

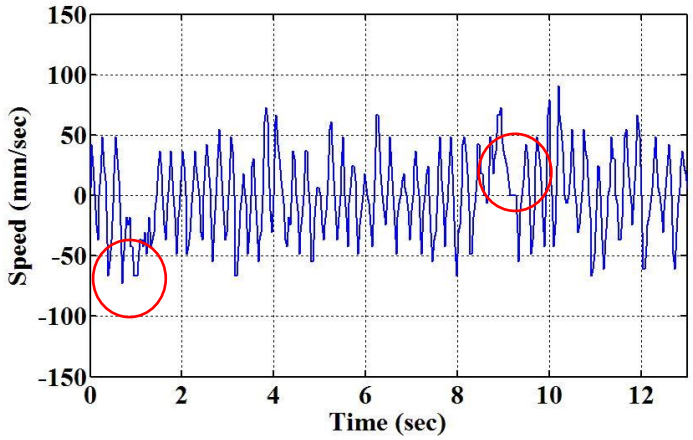


Figure 5.31: ***Travel speed while adding and removing the extra load at the back of the top layer of the robot***

5.8 Summary

In this chapter, the fuzzy logic control method was employed in order to control the movements of the robot and to maintain the balance state. Due to the non-linear nature of the two-wheeled robot and the coupling between the stats of the system that lead to uncertain mathematical model, a PD-type fuzzy logic controller has been designed to replace the internal control loop of the system which adjusts the input of the motors in a way to keep the two-wheeled robot balanced.

The controller was implemented digitally on the microcontroller to be run in real-time. A new technique has been developed for the digital implementation of the fuzzy controller. This technique increases the resolution of the output and reduces the required memory size, as it is based on real-time calculation rather than lockup table. Experimental studies were carried out to evaluate the implementation of the fuzzy controller and the performance of the two-wheeled robot system. It has been shown that the controller successfully maintained the balance state of the robot even in the presence of external disturbances. Furthermore, an extra weight of payload was added to and removed from the system at different locations whilst balancing in order to analyze the response of the robot for this change. The experimental results proved that the controller reacted reasonably well to the extra payload.

CHAPTER 6

Conclusion and further work

6.1 Summary and conclusion

The objective of this study has been the realization of a new 3-DOF structure of two-wheeled robot system with a movable payload. Furthermore, to design and implement a suitable controller for the system that is simple yet efficient for balancing and manoeuvring the two-wheeled robot and coping with the external disturbances. The system has been developed from the basic two-wheeled robot, inverted pendulum over cart, with new modification of adding an actuator, which moves and lifts the payload to an extended height along the intermediate body. This configuration adds an additional degree of freedom to the two-wheeled robot and contributes to serve new mobility solution applications.

Chapter two provides a hardware and software description for the two-wheeled robot test rig which has been used in this research. The technical specifications of the various part of the robot have been present. Furthermore, two PC software have been developed in Microsoft visual basic to offer a GUI with the two-wheeled robot and to help the researcher/user in design, edit, and code the fuzzy controller of the robot.

The first chapters of the thesis treated the system of two-wheeled robot considering a fixed position of the payload, where the position of the payload assumed at static position along the intermediate body, the chassis of the robot, as presented in chapters 3, 4, and 5. Hence, a mathematical model of the robot has been developed based on Euler-Lagrange approach. Then, investigations were carried out on the effect of different system parameters such as wheels size, payload weight, and position on the system performance and the control strategy.

In chapter four, a controller has been developed for balancing the two-wheeled robot based on the classic PID control theory. Later in chapter five, the fuzzy control theory has been utilized for the development of a controller for the robot. New approach has been developed for the real-time implementation of the fuzzy controller. Such approach eliminates the needs for lookup tables and provides an instantaneous output for any controller input values. Both controllers have been implemented on the AVR 8-bit microcontroller hardware for real-time execution, and tested experimentally under various operating conditions to evaluate the response of the two-wheeled robot system. They provided adequate results and managed to maintain the balancing state of the robot.

In chapter six, a dynamic payload motion and position has been considered rather than dealing with different static positions of a payload attached to the two-wheeled robot. In order to implement this dynamic motion, a few modifications on the hardware of the robot were necessary. Consequently, development of the control algorithm has been also required. Therefore, a linear actuator has been developed and added to the two-wheeled robot in order to activate the payload to move along the intermediate body with appropriate speeds. Adding a new actuator to the system adds an additional degree of freedom along the intermediate body.

An improved tilt sensor with an enhanced sensor fusion technique realized by Kalman filter have been adopted for tilt angle calculation. New controller structure has been developed for the new design of the robot. This controller was a combination of a PID feedback and feedforward control strategies. The controller with the Kalman filter have been implemented on the Mbed 32-bit ARM microcontroller hardware for real-time execution. Tests and experiments have been conducted on the two-wheeled robot considering dynamic motion for the payload with different profiles. Also the system has been tested for three different weights of the payload. The results revealed that the two-wheeled robot system would consume less energy to stabilizes and balance the robot when the COM located at a higher position. Further experiments for balancing the robot on inclined surfaces have performed. These tests have been made on three inclination angles between 5 and 10 degrees. The results showed that the controller managed to balance the two-wheeled robot on inclined planes within acceptable error margins.

6.2 Recommendation for future work

The main objectives of this research have been to realize in real-time a new structure design of two- wheeled robot with an additional degree of freedom and to investigate an appropriate control algorithm for the system. During the research period, there were few constrains, time limitations, and hardware failures, due to which several development ideas were highlighted and several issues could not be investigated. Therefore, some of these ideas were left for future work and they are described in the following sections.

6.2.1 Further hardware development

Several improvements may be considered to improve the robot's hardware. This improvement may give a clear view for scaling up the design to be used in practical application.

6.2.1.1 Microcontroller board

Development board that supports real-time OS such as the TS-7200 SBC, which provides high processing speed and real-time Linux OS, may be considered to implement the control system of the two wheeled robot. This feature may help in implementing an online self-tuning, and controllers been upgraded to more powerful one. Having a multitasking operating system implies that other processes can be implemented alongside the main balancing control process such as obstacle avoidance, path planning, and human robot interaction, etc..

6.2.1.2 Motors and payload actuator

Although the issue of weak gears of the motors has been solved with new DC motors with metal gears; another issue was noted which was the motor backlash, the clearance or lost motion in the motor's shaft caused by the gaps between the gear parts. This can be reduced or even eliminated by using motors with planetary gearing rather than the normal spur gearing.

The travelling speed of the payload actuation unit is limited to the maximum speed of the stepper motor and the pitch size of the threaded screw. Hence a new design of the linear actuation unit may be investigated to provide a higher travelling speed for the payload.

6.2.1.3 Sensors improvement

Other types of sensors could also be added to the robot in order to make it more autonomous. Eventually, any improvement of the sensors of the two-wheeled robot system leads to enhancement in the performance of the system. Further development could be carried out on either adding more sensors to get additional information or developing the current sensors of the system. An example on the additional sensor is sonar of range finder, visual sensors like camera, GPS, and magnetometer which can be integrated with the MPU 6050 sensor.

Other development on the sensing technique is recommended, for instance the travelled distance of the two-wheeled robot is calculated from the wheels encoder; this may lead to false distance measurement specially when there is slipping in the wheels. A laser motion sensor, similar to the mouse sensor, can be included for measuring the travelled distance in both X and Y axis.

Further research on measuring the inclination angle of the surface may be considered through adding additional sensors. This would be supportive in developing an adaptive control strategy to move from flat to inclined surfaces and vice versa.

6.2.2 Further research

- In this research, the development and realization of the fuzzy control technique for two-wheeled robots has been presented. Tuning of the controller parameters such as MSF distribution and shape, may be investigated to improve system performance. Other intelligent control algorithm may be considered to develop the controller of the robot such as neural network.

- Adaptive or auto-tune control technique may be adopted to improve the performance of the new structure two-wheeled robot during the dynamic motion and weight of the payload. A separate decoupled control algorithm for the two-wheeled robot system, can be developed in order to achieve robust steering control and overcome the effects of wheels slip.

- In this research, the use of non-tuned parameters was minimized as possible. However, offline tuning of some of the parameters would be beneficial in speed up the tuning of the controller. Therefore, system identification technique may be adopted in order to build a model for the system based on the experimental input output data. Such model will facilitate optimizing the parameters of the system for a better response by choosing the appropriate optimization method.

REFERENCES

Abeygunawardhana, P. and Toshiyuki, M. (2007). "*Stability Improvement of Two Wheel Mobile Manipulator by Real Time Gain Control Technique"*. Second International Conference on Industrial and Information Systems, ICIIS, Penadeniya, Sri Lanka, 8-11 Aug. 2007, pp.79-84.

Abeygunawardhana, P. and Toshiyuki, M. (2008). "*Environmental Interaction of Two Wheeled Mobile Manipulator by Using Reaction Torque Observer"*. 10th IEEE International Workshop on Advanced Motion Control, AMC '08, Trento, 26-28 Mar. 2008, pp.348-353.

Ahmad, S., Tokhi, M.and Toha, S. (2009). "*Genetic Algorithm Optimisation for Fuzzy Control of Wheelchair Lifting and Balancing"*. Third UKSim European Symposium on Computer Modeling and Simulation, Athens, 25-27 Nov. 2009, pp.97-101.

Almeshal, A. M., Tokhi, M. and Goher, K. (2011). "*Modelling of Two-Wheeled Robotic Wheelchair with Moving Payload"*. The 14th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, Paris, France, 6-8 Sept. 2011.

Angeli, D. (2001). "*Almost Global Stabilization of the Inverted Pendulum Via Continuous State Feedback*." Automatica, vol. **37**(7), pp.1103-1108, 2001.

Baloh, M. and Parent, M. (2003). "*Modeling and Model Verification of an Intelligent Self-Balancing Two-Wheeled Vehicle for an Autonomous Urban Transportation System"*. The Conference on Computational Intelligence, Robotics, and Autonomous Systems, Singapore, 15 Dec. 2003, pp.1-7.

Bohn, C. and Atherton, D. P. (1995). "*An Analysis Package Comparing PID Anti-Windup Strategies*." Control Systems, IEEE, vol. **15**(2), pp. 34-40, 1995.

Chen, C.-H., Jean, J.-H. and Xu, D.-X. (2011). "*Application of Fuzzy Control for Self-Balancing Two-Wheel Vehicle"*. International Conference on Machine Learning and Cybernetics (ICMLC), Guilin, 10-13 Jul. 2011, pp.1204-1209.

Chou, C.-H. and Lu, H.-C. (1993). "*Design of a Real-Time Fuzzy Controller for Hydraulic Servo Systems*." Computers in Industry, vol. **22**(2), pp. 129-142, 1993.

Coelho, V., Liew, S., Stol, K. and Liu, G. Y. (2008). "*Development of a Mobile Two-Wheel Balancing Platform for Autonomous Applications"*. 15th International Conference on Mechatronics and Machine Vision in Practice (M2vip), Auckland, New-Zealand, 2-4 Dec. 2008, pp.552-557.

Farooq, U., Asad, M. U., Hanif, A., Hasan, K.and Amar, M. (2012). "*Design and Implementation of a Fuzzy Logic Controller for Two Wheeled Self Balancing Robot*." Advanced Materials Research, vol. **403-408**, pp.4918-4925, 2012.

Feng, T., Liu, T., Wang, X., Xu, Z., Zhang, M. and Han, S.C. (2011). "*Modeling and Implementation of Two-Wheel Self-Balancing Robot Equipped with Supporting Arms"*. 6th IEEE Conference on Industrial Electronics and Applications (ICIEA), Beijing, China, 21-23 Jun. 2011, pp.713-718.

Fukushima, H., Muro, K. and Matsuno, F. (2015). "*Sliding-Mode Control for Transformation to an Inverted Pendulum Mode of a Mobile Robot with Wheel-Arms*." IEEE Transactions on Industrial Electronics, , vol. **62**(7), pp. 4257-4266, 2015.

Goher, K., Ahmad, S. and Tokhi, O. M. (2010). "*A New Configuration of Two Wheeled Vehicles: Towards a More Workspace and Motion Flexibility"*. 4th Annual IEEE Systems Conference, 2010, San Diego, CA, 5-8 Apr. 2010, pp. 524-528.

Goher, K. and Tokhi M. (2009). "*GA-Optimised Steering and Position Control a Two Wheeled Vehicle with an Extended Rod – a Simulation Study"*. The 12th International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines, Istanbul, Turkey, 9-11 Sept. 2009, pp. 66-74.

Goher, K. M. K. and Tokhi, M. O. (2010). "*Genetic Algorithm Based Modeling and Control of a Two Wheeled Vehicle with an Extended Rod, a Lagrangian Based Dynamic Approach"*. IEEE 9th International Conference on Cybernetic Intelligent Systems (CIS), Reading, UK, 1-2 Sept. 2010.

Goher, K. M. K. and Tokhi, M. O. (2010). "*A Two-Wheeled Vehicle with an Extended Rod, Mechanical Design Clues and Dynamic Modelling"*. IEEE 9th International Conference on Cybernetic Intelligent Systems (CIS), Reading, UK, 1-2 Sept. 2010.

Grant, R. (2009). *"Tip two wheeled balancing robot: Two-Wheeled Balancing Robot Project"*. Fun with basic robot, 11 Apr. 2009, accessed on Oct. 2015, web

link: http://basicrobot.blogspot.co.uk/2009/04/two-wheeled-balancing-robot-project-2.html .

Grasser, F., D'Arrigo, A., Colombi, S. and Rufer, A. C. (2002). "*Joe: A M|Obile, Inverted Pendulum*." IEEE Transactions on Industrial Electronics, vol. **49**(1), pp. 107-114, 2002.

Hassenplug, S. (2002). "*Steve's LegWay:A LEGO two wheeled robot*", accessed on Oct. 2015, web link: http://www.teamhassenplug.org/robots/legway/ .

Hata, H. and Takimoto, T. (2014). "*Development of the Portable Two-Wheeled Inverted Pendulum Type Personal Vehicle"*. 14th International Conference on Control, Automation and Systems (ICCAS), Gyeonggi-do, Korea, 22-25 Oct. 2014, pp.1610-1613.

Higgins, W. (1975). "*A Comparison of Complementary and Kalman Filtering*." IEEE Transactions on Aerospace and Electronic Systems, vol. **AES-11**(3), pp. 321-325, 1975.

Huang, C. N. (2010). "*The Development of Self-Balancing Controller for One-Wheeled Vehicles*." Engineering, vol. **2**, pp. 212-219, 2010.

Hyung-Jik, L. and Seul, J. (2009). "*Gyro Sensor Drift Compensation by Kalman Filter to Control a Mobile Inverted Pendulum Robot System"*. IEEE International Conference on Industrial Technology ICIT 2009, Gippsland, VIC, 10-13 Feb. 2009.

Hyung, J. L., Hyun Wook, K. and Seul, J. (2010). "*Development of a Mobile Inverted Pendulum Robot System as a Personal Transportation Vehicle with Two Driving Modes :Transbot"*. World Automation Congress (WAC), 2010, Kobe, 19-23 Sept. 2010.

Jean, J. H. and Wang, C. K. (2009). "*Design and Implementation of a Balancing Controller for Two-Wheeled Vehicles Using a Cost-Effective MCU"*. Proceedings of 2009 International Conference on Machine Learning and Cybernetics, Baoding, 12-15 July 2009, pp. 3329-3334.

Jingtao, L., Xueshan, G., Qiang, H., Qinjun, D. and Xingguang, D. (2007). "*Mechanical Design and Dynamic Modeling of a Two-Wheeled Inverted Pendulum Mobile Robot"*. IEEE International Conference on Automation and Logistics, Jinan, China, 18-21 Aug. 2007, pp. 1614-1619.

Junfeng, W. and Shengwei, J. (2011). "*T-S Adaptive Neural Network Fuzzy Control Applied in Two-Wheeled Self-Balancing Robot"*. 6th International Forum on Strategic Technology (IFOST 2011), Harbin, Heilongjiang, 22-24 Aug. 2011, pp.1023-1026.

Junfeng, W. and Wanying, Z. (2011). "*Design of Fuzzy Logic Controller for Two-Wheeled Self-Balancing Robot"*. 6th International Forum on Strategic Technology (IFOST 2011), Harbin, Heilongjiang, 22-24 Aug. 2011, pp. 1266-1270.

Junfeng, W., Yuxin, L. and Zhe, W. (2011). "*A Robust Control Method of Two-Wheeled Self-Balancing Robot"*. 6th International Forum on Strategic Technology (IFOST 2011), Harbin, Heilongjiang, 22-24 Aug. 2011, pp. 1031-1035.

Jung, S. and Kim, S. (2007). "*Hardware Implementation of a Real-Time Neural Network Controller with a DSP and an FPGA for Nonlinear Systems*." IEEE Transactions on Industrial Electronics , vol. **54**(1), pp. 265-271, 2007.

Jung, S. and Kim, S. S. (2008). "*Control Experiment of a Wheel-Driven Mobile Inverted Pendulum Using Neural Network*." IEEE Transactions on Control Systems Technology, vol. **16**(2), pp. 297-303, 2008.

Kalman, R. E. (1960). "*A New Approach to Linear Filtering and Prediction Problems*." Transactions of the ASME – Journal of Basic Engineering, vol. **Series D**(82), pp. 35-45, 1960.

Kamen, D. (2001). The Segway® Personal Transporter (PT), the First Self-Balancing, Zero Emissions Personal Transportation Vehicle.

Kamen, D. and Johnson and Johnson's Independence Technology©, (2001). Stair-Climbing Wheelchair - iBOT® Mobility System. accessed on Oct. 2015, web link: https://en.wikipedia.org/wiki/IBOT .

Kim, S. and Jung, S. (2006). "*Hardware Implementation of a Neural Network Controller with an MCU and an FPGA for Nonlinear Systems*." INTERNATIONAL JOURNAL OF CONTROL AUTOMATION AND SYSTEMS, vol. **4**(5), pp. 567, 2006.

Kuindersma, S. R., Hannigan, E., Ruiken, D. and Grupen, R. A. (2009). "*Dexterous Mobility with the Ubot-5 Mobile Manipulator"*. International Conference on Advanced Robotics, ICAR 2009, Munich, Germany, 22-26 June 2009.

Leavitt, J., Sideris, A. and Bobrow, J. E. (2006). "*High Bandwidth Tilt Measurement Using Low-Cost Sensors*." IEEE/ASME Transactions on Mechatronics , vol. **11**(3), pp. 320-327, 2006.

Lee, H. and Jung, S. (2012). "*Balancing and Navigation Control of a Mobile Inverted Pendulum Robot Using Sensor Fusion of Low Cost Sensors*." Mechatronics, vol. **22**(1), pp. 95-105, 2012.

Li, C., Dai, F., Li, F., Wang S., Gao, X. and Li, K. (2011). "*Development of a Dual-Mode Mobile Robot System for Practical Applications"*. International Conference on Mechatronics and Automation (ICMA 2001), Beijing, China, 7-10 Aug. 2011, pp. 1485-1490.

Li, Z. and Yang, C. (2011). "*Neural-Adaptive Output Feedback Control of a Class of Transportation Vehicles Based on Wheeled Inverted Pendulum Models*." IEEE Transactions on Control Systems Technology, vol. **20**(6), pp. 1583 - 1591, 2011.

Mamdani, E.H., (1974). *"Application of fuzzy algorithms for control of simple dynamic plant*". In Proceedings of the Institution of Electrical Engineers vol. **121**(12), pp. 1585-1588. IET Digital Library.

Marzi, H. (2005). "*Multi-Input Fuzzy Control of an Inverted Pendulum Using an Armature Controlled DC Motor*." Robotica, vol. **23**(06), pp. 785-788, 2005.

Miasa, S., Al-Mjali, M., Ibrahim, A. H. and Tutunji, T. A. (2010). "*Fuzzy Control of a Two-Wheel Balancing Robot Using Dspic"*. 7th International Multi-Conference on Systems Signals and Devices (SSD), Amman, Jordan, 27-30 June 2010.

Murakami, T. and Sasaki, K. (2009). "*A Motion Control for Two-Wheel Inverted Pendulum Type of Mobile Manipulator*." IEEJ Transactions on Electrical and Electronic Engineering, vol. **4**(2), pp. 192-198, 2009.

Nakagawa, C., Nakano, K., Suda, Y. and Hirayama, Y. (2011). "*Stability of the Two-Wheeled Inverted Pendulum Vehicle Moved by Human Pedaling*." Journal of System Design and Dynamics, vol. **5**(3), pp. 389-402, 2011.

Nawawi, S. W., Ahmad, M. N. and Osman, J. H. S. (2008). "*Real-Time Control of a Two-Wheeled Inverted Pendulum Mobile Robot*." International Journal of Computer, Information, and Systems Science, and Engineering, vol. **2**(1), pp. 70-76, 2008.

Ogata, K. (2002). Moder Control Engineering, 4th edition, pearson education international.

Passino, K. M., Yurkovich, S. and Reinfrank, M. (1998). Fuzzy Control, California, USA, Addison Wesley Longman, Inc.

Pathak, K., Franch, J. and Agrawal, S. K. (2005). "*Velocity and Position Control of a Wheeled Inverted Pendulum by Partial Feedback Linearization*." IEEE Transactions on Robotics , vol. **21**(3), pp. 505-513, 2005.

Petrov, P. and Parent, M. (2010). "*Dynamic Modeling and Adaptive Motion Control of a Two-Wheeled Self-Balancing Vehicle for Personal Transport"*. 13th International IEEE Conference on Intelligent Transportation Systems (ITSC 2010), Madeira Island, Portugal, 19-22 Sept. 2010, pp. 1013-1018.

Prasad, L. B., Tyagi, B. and Gupta, H. O. (2011). "*Optimal Control of Nonlinear Inverted Pendulum Dynamical System with Disturbance Input Using PID Controller &Amp; Lqr"*. IEEE International Conference on Control System, Computing and Engineering (ICCSCE2011), Sivakasi, India, 25-27 Nov. 2011, pp. 540-545.

Ren, T.-J., Chen, T. C. and Chen, C. J. (2008). "*Motion Control for a Two-Wheeled Vehicle Using a Self-Tuning Pid Controller*." Control Engineering Practice, vol. **16**(3), pp. 365-375, 2008.

Rukidi, P. E., Fernandes, J. M. and Phillips, G. (2014). "*Development of a Two-Wheel Balancing Robot Using the Stm32f3-Discovery Board as an Educational Platform for Traditional and Modern Control Schemes"*. Proceedings of the 2014 PRASA, RobMech and AfLaT International Joint Symposium, Cape Town, South Africa, 27-28 November 2014.

Salerno, A. and Angeles, J. (2007). "*A New Family of Two-Wheeled Mobile Robots: Modeling and Controllability*." IEEE Transactions on Robotics , vol. **23**(1), pp. 169-173, 2007.

Seong Hee, J. and Takahashi, T. (2007). "*Wheeled Inverted Pendulum Type Assistant Robot: Inverted Mobile, Standing, and Sitting Motions"*. IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007, San Diego, CA, USA, 29 Oct. - 2 Nov. 2007, pp. 1932-1937.

Solis, J., Nakadate, R., Yoshimura, Y., Hama, Y. and Takanishi, A. (2009). Development of the Two-Wheeled Inverted Pendulum Type Mobile Robot Wv-2r for Educational Purposes. IEEE-Rsj International Conference on

Intelligent Robots and Systems. Louis, USA, 11-15 Oct. 2009, pp. 2347-2352.

Takahashi, Y., Ishikawa, N. and Hagiwara, T. (2003). "*Soft Raising and Lowering of Front Wheels for Inverse Pendulum Control Wheel Chair Robot"*. Proceedings. 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS 2003), Las Vegas, Nevada, 27-31 Oct. 2003, pp. 3618-3623

Takahashi, Y. and Kohda, M. (2005). "*Human Riding Experiments on Soft Front Wheel Raising of Robotic Wheelchair with Inverse Pendulum Control"*. IEEE International Conference on Industrial Technology, ICIT 2005, Hong Kong, 14-17 Dec. 2005, pp. 266-271.

Takahashi, Y., Ogawa, S. and Machida, S. (1999). "*Front Wheel Raising and Inverse Pendulum Control of Power Assist Wheel Chair Robot"*. The 25th Annual Conference of the IEEE Industrial Electronics Society, IECON '99 San Jose, CA, USA, 1999, pp. 668-673

Takahashi, Y., Takagaki, T., Kishi, J. and Ishii, Y. (2001). "*Back and Forward Moving Scheme of Front Wheel Raising for Inverse Pendulum Control Wheel Chair Robot"*. Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation, 2001, Seeoul, South Korea, 21-26 May 2001, pp.3189-3194.

Tsai, C. C., Huang, H. C. and Lin, S. C. (2010). "*Adaptive Neural Network Control of a Self-Balancing Two-Wheeled Scooter*." IEEE Transactions on Industrial Electronics, , vol. **57**(4), pp. 1420-1428, 2010.

Vermeiren, L., Dequidt, A., Guerra, T. M., Rago-Tirmant, H. and Parent, M. (2011). "*Modeling, Control and Experimental Verification on a Two-Wheeled Vehicle with Free Inclination: An Urban Transportation System*." Control Engineering Practice, vol. **19**(7), pp. 744–756, 2011.

Wang, Q. and Fang, J. (2014). "*Fuzzy Immune PDAlgorithm Applied in the Self-Balancing Two-Wheeled Robot"*. 8th International Conference on Future Generation Communication and Networking (FGCN 2014), Haikou, China, 20-23 Dec. 2014, pp. 112-115.

Zadeh, L.A., (1965). Fuzzy sets. Information and control, **8**(3), pp.338-353.

Ziegler, J.G. and Nichols, N.B., (1942). Optimum settings for automatic controllers. trans. ASME, **64**(11).

Appendix A

Specifications of the two wheeled robot hardware

A.1 Introduction

In this section, a description of the main mechanical and electrical parts of the two-wheeled robot test rig is presented. These parts include the wheels and motors, the payload actuation unit of the robot, sensors, motor drivers, and the microcontroller development boards.

A.1.1 DC motors

Two permanent magnet DC motors, EMG30, are used to drive the wheels of the robot. These motors run at 12 Volt and provide a maximum no load speed of 216 RPM through 1/30 reduction gear. Each of these motors has a magnetic encoder connected to its shaft and generates 360 pulses per wheel revolution. The shaft of motors was directly coupled with a 100mm wheel. Later in chapter six, these motors will be replaced with the Pololu DC motors. Which are equipped with metal gear head and provide higher torque as shown in Figure A.1. The technical specifications of the DC motors used in the two-wheeled robot, are listed in Table A.1.

Table A.1: ***DC motor specifications***

Specification	EMG30	Pololu	Unit
Rated voltage	12	12	V
Rated torque	1.5	8	Kg/cm
No load speed	216	350	rpm
No load current	0.15	0.3	A
Stall current	2.5	5	A
Gear ratio	1:30	Metal 1:30	--
Encoder counts per output shaft	360	1920	pulse
Size	30D x 86L	37D x 64L	mm
Weight	0.18	0.21	kg

a. EMG30 motor ***b. Pololu motor***

Figure A.1: ***DC motors of the two-wheeled robot prototype***

A.1.2 Payload actuation motor

Figure A.2 shows the structure diagram of the payload actuation unit. It consists of two co-axial parts and a non-captive stepper motor, with its shaft attached to the payload base. The maximum travelling length of the payload actuation unit is around 20 cm as the stepper motor is equipped with a 25cm thread screw shaft as shown in Figure A.3. The bipolar nema17 was used and according to its technical specifications, listed in Table A.2, the motor is able to push a payload with a maximum weight of 4 kg at maximum travelling speed of 10 mm per second.

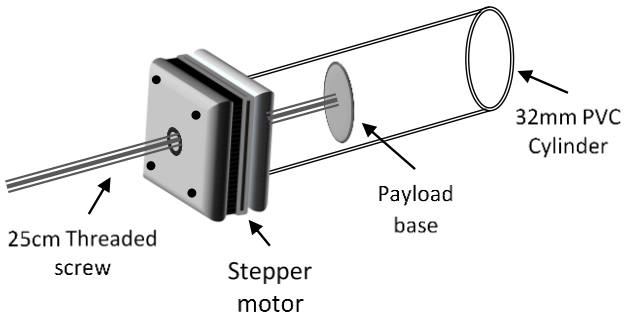


Figure A.2: ***Payload actuation unit structure***

Figure A.3: ***Non-captive stepper motor***

Table A.2 ***Stepper motor specifications***

Specification	Value	Unit
Step angle	1.8	°
Step distance	0.01	mm
Torque	0.21	Nm
Pushing force	0.4	N
Voltage	12	V
Rate current	0.4	A
Max travel speed	10	mm/sec
Max travel distance	220	mm
Size	43 x 43 x 40	mm
Weight	0.18	kg

A.2 Electrical and electronic parts

A.2.1 Microcontrollers and module boards

A.2.1.1 AVR development board

The AVR board uses the ATmega32 microcontroller by Atmel, which is 8-bit clocked by an external oscillator at 16MHz. The memory capabilities of the board include 32 K bytes of flash program memory, 1 K bytes of EEPROM and 2 K bytes internal SRAM. The peripherals include 32 digital input output lines of which there are eight 10-bit analogue/digital converter ADC channels, four pulse width modulation PWM channels, a serial peripheral interface SPI bus, and two-wire serial interface I^2C. This development board provides headers for programming functions such as In-circuit Serial Programming ISP as well as a J-tag interface. Figure A.4 shows the AVR development board.

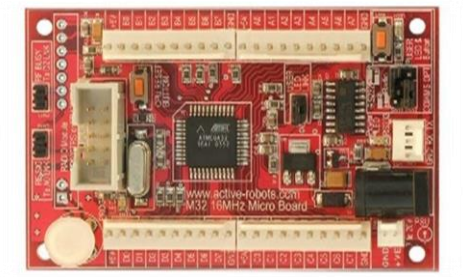


Figure A.4: ***AVR development board***

A.2.1.2 TS-7200 board

The TS-7200 is a compact full featured single board computer SBC based on the Cirrus EP3902 ARM9 CPU, shown in Figure A.5. The EP9302 has a CPU speed of 200 MHz with a memory management unit that supports high-level real time embedded operating systems such as Linux, Windows, and others. It has 32Mbytes programmable flash memory, and two serial ports. Also, it has twenty digital I/O pins, eight channels 12-bit ADC, True IDE Compact Flash socket (for additional memory), two USB 2.0 compatible ports, 10/100 Megabit Ethernet port, watchdog timer, Alphanumeric LCD and Matrix keypad interfaces and PC/104 expansion bus.

The ARM9 32-bit architecture, with a five-stage pipeline, delivers very impressive performance at very low power. It has a 16 KB instruction cache and a 16 KB data cache to provide zero-cycle latency to the current program and data, or it can be locked to guarantee no-latency access to critical sections of instructions and data. The ARM9 compressed Thumb instruction set can be used to provide a higher code density and lower Flash storage requirements for applications with instruction-memory size restrictions. Therefore, the TS-7200 SBC was considered in the improvement of the hardware of the robot as will be discussed later in chapter six.

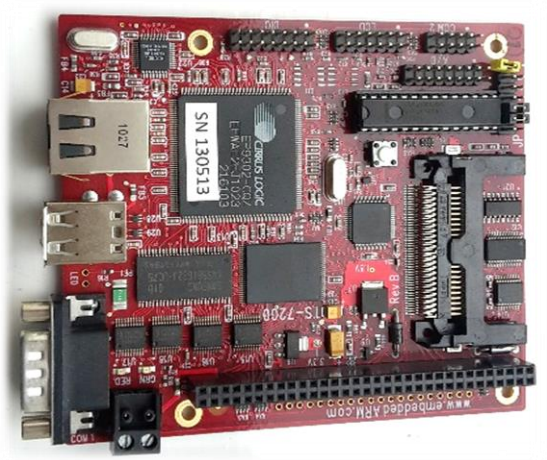


Figure A.5: ***TS-7200 single board computer***

A.2.1.3 The Mbed development board

The Mbed is a platform and operating system for internet-connected devices based on 32-bit ARM Cortex-M microcontrollers. The project is collaboratively developed by ARM and its technology partners. There are various hardware demo boards for the Mbed platform, with the first being the original Mbed Microcontroller board. The Mbed Microcontroller Board, marketed as the "mbed NXP LPC17680", is a development board based on an NXP microcontroller with an ARM Cortex M3 core, running at 96 MHz. Thw Mbed has 512 KB programmable flash memory, with 64 KB of RAM, as well as several interfaces such as serial ports, up to 3 ports, Ethernet, USB device/host, CAN, two SPI buses,

and two I²C buses. It has 25 general purpose digital I/O lines of which six 12-bit ADC channels, six PWM outputs, and an analogue out channel as depicted in Figure A.6.

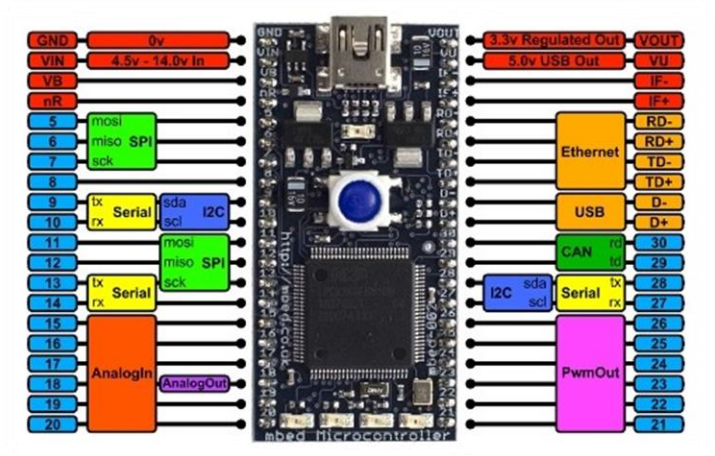


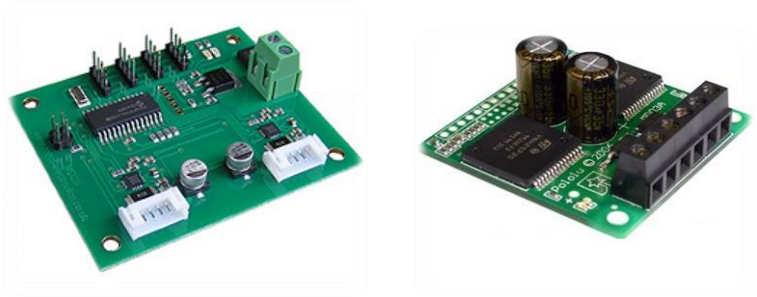
Figure A.6: ***Mbed NXP LPC17680 development board***

A.2.2 DC Motor driver module

In order to run the motors with variable speed and direction, a DC motor driver board was utilized such as the MD-25 board, and the Pololu MD03A.

The MD-25 is a dual 2.8A H-Bridge DC motor controller. This board can be programmed over serial or I^2C bus interface and it supports two operation modes common and individual motor control. The board can report the current, voltage as well as the encoder information of each motor. The MD-25 board represents a closed loop speed controller for the motors with a programmable acceleration as it receives the control commands over the interface bus and the encoder's information as a feedback.

The Pololu MD03A is a compact high power dual motor driver board which has continuous diving current of 9A per channel. This board takes PWM input, up to 10 KHz, to control the speed of the motor. The rotation direction of the motor can be set through a digital input. This driver board was used with the Pololu DC motors. Figure A.7 shows the DC motor driver boards which have been used in this research.



a. MD-25 *b. Pololu MD03A*

Figure A.7: ***DC motor driver boards***

A.2.3 Stepper motor driver module

A stepper motor driver was used to drive the stepper motor of the payload actuation unit. It is based on the DRV8825 chip on a small board, as in Figure A.8; designed to driver bipolar motors through two digital inputs for direction and step. It can provide up to 2.2A per coil with adjustable current limit. It supports input voltage up to 45 volt and interfaces with 3-5 volt inputs directly. The DRV8825 offers six different micro-stepping modes: full-step, half-step, 1/4-step, 1/8-step, 1/16-step, and 1/32-step.

Figure A.8: ***DRV8825 stepper motor driver board***

A.2.4 Bluetooth module

A wireless data communication link was realized by the HC-06 serial over a Bluetooth module, shown in Figure A.9, which is responsible for relaying all serial communication over wireless RF link using the Bluetooth protocol. This module supports data transfer rate up to 115kbps over a distance of up to 30ft and authenticates using a user changeable pairing code. In such way, the main

controller of the two-wheeled robot can communicate with any PC equipped with a Bluetooth dongle, where navigation commands can be sent to the robot from the PC and debugging information can be read and recorded.

Figure A.9: ***HC-6 serial Bluetooth model board***

A.2.5 Sensors

In order to maintain the robot in a balancing state, it is equipped with three main sensors, encoders, accelerometer, and gyroscope. However it can be equipped with other types of sensor for other functionality such as a digital compass for navigation or range finder for obstacle avoidance.

A.2.5.1 Wheels encoder

In order to measure the angular displacement and angular velocity of the wheels of the robot, each wheel was equipped with an incremental encoder that gives 360 pulses per wheel rotation.

A.2.5.2 Accelerometer

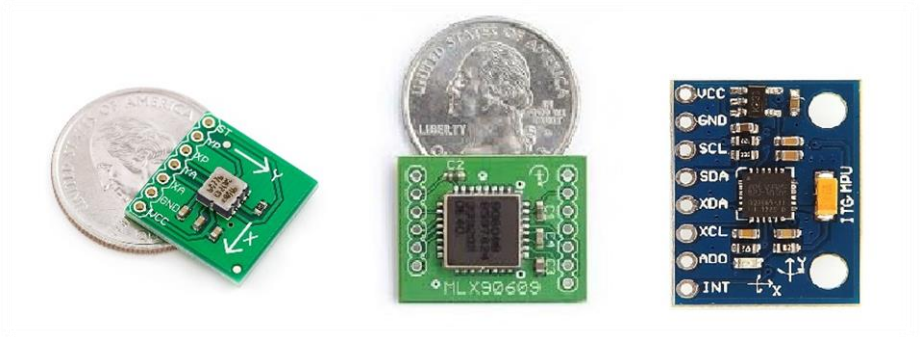
The ADXL213 accelerometer sensor by Analogue Devices was used. This is a small low power 2-axis accelerometer. It generates an analogue output signal for each axis in the range of 0-5 Volts, which is proportional to acceleration of ±1.2g with 300 mV/g sensitivity. The ADXL213 was utilized for tilt angle sensing of the robot by measuring the static acceleration of gravity for a defined orientation of the sensor.

A.2.5.3 Gyroscope

The MXL90609 gyroscope sensor by Melexis is used, which is a single axis Microelectromechanical systems (MEMS) gyroscope. This sensor produces an output proportional to the rotational speed around the sensing axis in the range of ±150 °/sec. It features low drift and programmable bandwidth with two choices of output reading, either analogue output in the range of 0-5 Volt with 13.33 mV/ °/sec sensitivity or digital output, using internal 11-bit ADC, through an SPI connection.

A.2.5.4 MPU 6050

The InvenSense MPU-6050 sensor is a dual MEMS sensors in a single chip, as it combines a MEMS 3-axis gyroscope and a 3-axis accelerometer. It provides an accurate measurement for precision tracking of both fast and slow motions. It features a user-programmable gyro full-scale range of ±250, ±500, ±1000, and ±2000°/sec and a user-programmable accelerometer full-scale range of ±2g, ±4g, ±8g, and ±16g. These measurements are internally calibrated and transferred through I^2C bus of the sensor. The sensor can even access external magnetometers or other sensors through an auxiliary master I2C bus, allowing the devices to gather a full set of sensor data without intervention from the system processor. Furthermore, the built in Digital Motion Processor™ (DMP™) is capable of processing complex 9-axis Motion Fusion algorithms. The MPU-6050 is considered in the tilt angle measurement improvement in chapter six. Figure A.10 shows the tilt angle sensor modules used in this work.



a. ADXL213 *b. MXL90609* *c. MPU 6050*

Figure A.10: ***Tilt sensor accelerometer and gyroscope modules***