بسم الله الرحمن الرحيم

Sudan University of Science and Technology

College of Graduate Studies

Designing and implementation of PID controller robotic arm

تصميم وتطبيق المتحكم التناسبي التفاضلي التكاملي لزراع ربوت

A Dissertation submitted in partial fulfillment for the requirementof M.Sc. Degree in mechatronics engineering

Prepared by:

Hosham Wahballa Abdalla Mahamed

Supervised by:

Dr. Ahmed Abdelrahaman Abdalla

March 2018

الأيـــه

بسم الله الرحمن الرحيم

قال تعالي

(اقْرَأْ بِاسْمِ رَبِّكَ الَّذِي خَلَقَ{1} خَلَقَ الْإِنسَانَ مِنْ عَلَقٍ {2} اقْرَأْ وَرَبُّكَ الْأَكْرَمُ{3} الَّذِي عَلَّمَ بِالْقَلَمِ{4} عَلَّمَ الْإِنسَانَ مَا لَمْ يَعْلَمْ{5})

صدق الله العظيم

Dedication

With heart of full happiness, it's my pleasure to dedicate this study

To my

Parents

Brothers and friends

To all our distinguished teachers

Without whom none of my success would be possible

Acknowledgements

Above all, all praise is due to Allah almighty in the beginning and the end, who gave me health, strength, and patience to finish this work.

I wish to express my sincere appreciation to my supervisor **Dr. Ahmed Abdelrahaman Abdallah** for supervision, comments and guidance in order to make sure I can gain as much more experience and knowledge to complete my research.

Finally, I would like to thank all my friends and my colleagues in Sudan University, College of Post Graduate. Also I would like to thank all my teachers and colleagues in Electrical and Computer Department_ Faculty of Engineering_Krary University, and sharing ideas during the progress of this thesis, thank you all times without number.

Abstract

Designing and controlling a robotic arm with great performance is one of the fields of interest in many applications. This thesis is concerned with the problems control of robot arm (mentor) using Proportional Integral Derivative (PID) controller for axis1 (shoulder) and axis2 (elbow). The research work was undertaken in the following developmental stages; first stage, estimation of the transfer functions of Direct current (DC) motors actuators of tow axis's using system identification data driven control method depend on arduino Uno and support package used to design simulinks in real time. Second stage, PID Controller is designed by using MATLAB/SIMULINK and applied it to above tow DC motors. Third stage, the simulation is performed to compare between system before and after PID controller, this is done through the simulation on the computer using MATLAB/SIMULINK. Fourth stage, implementing the PID controller designed in second stage to tow axis of robot, and test the system in real time to compares desired angle form and actual angle form response of DC motor.

المستخلص

ان عمليات التصميم والتحكم ذو الاداء العالي لزراع الربوت يعتبر من اهم المجالات في معظم التطبيقات المنتشرة اليوم في مجال الربوت. هذه الرسالة البحثية تهدف لعملية التصميم والتحكم في زراع ربوت(يسمي منتور) وذلك باستخدام المتحكم التناسبي التفاضلي التكاملي لمحورين اساسين في الربوت (الكتف والمرفق). تم العمل في هذا البحث عبر عدة مرعلاحل: المرحلة الاولي هي التعرف علي الموتورات في المحورين 1و2 الموجودة في هذا الربوت وايجاد الدوال التحويلية لها وذلك باستخدام طريقة تسمي التحكم في الحركة بيانيا وذلك باستخدام متحكمة (الاردينو انو) وبرامج خاصة تم تصميمها في الماتلاب وذلك باستخدام الزمن الحقيقي. المرحلة التانية هي تصميم الحاكم التناسبي التفاضلي التكاملي باستخدام برنامج الماتلاب وتطبيقها علي الموتورات المختارة من الربوت. المرحلة الثالثة هي عمل محاكاة للنظام قبل وبعد تصميم المتحكم التناسبي التفاضلي التكاملي وتتم هذه المحاكاة في الكمبيوتر بواسطة برنامج الماتلاب. المرحلة الرابعة والاخيرة هي تطبيق المتحكمة عمليا علي حركة الموتورات اعلاه وعمل الاختبار لها في الزمن الحقيقي بعمل مقارنة بين الحركة المطلوبة والحركة الحقيقية لاستجابة للموتور.

List of Figures

List of Table

List of Abbreviations

DC	Direct Current
PID	Proportional Integral Derivative
DOF	Degree Of freedom
GUI	Graphical User Interface
R.U.R	ROSSUM'S Universal Robots
P	Proportional
PI	Proportional-Integral
PD	Proportional-Derivative
Z-N	Ziegler-Nichols
S-S error	Steady state error
NSS	Not Steady State
CPU	Central Processing Unit
PWM	Pulse Width Modulation
EEPROM	Electrical Erasable Programmable Read Only Memory
SRAM	Static random access memory
USB	Universal Serial Bus
ADC	Analog to Digital Converter
DDC	Data Driven Control
ICSP	In-Circuit Serial Programming
AC	Alternative Current
DTL	Diode Transistor Logic
TTL	Transistor-Transistor Logic
MATLAB	MATrix LABoratory
DES	Desired
ACT	Actual
IDDATA	IDentification DATA
TF	Transfer Function

Table of Contents

Chapter one
Introduction

1.1 Overview

Industrial and commercial systems with high efficiency and great performance have taken advantages of robot technology. Large number of control researches and numerous control applications were presented during the last years, concentrated on control of robotic systems. Robot manipulator field is one of the interested fields in industrial, educational and medical applications. It works in unpredictable, hazard and in hospitable circumstances which human cannot reach. For example, working in chemical or nuclear reactors is very dangerous, while when a robot instead human it involves no risk to human life. Therefore, modeling and analysis of the robot manipulators and applying control techniques are very important before using them in these circumstances to work with high accuracy.

When the need arises for linear motion or positioning there are many choices. One can use a linear motor. Some of the common linear motors are stepper, DC motor, synchronous, hybrid, induction motors. The DC servo motor was one of the first linear motors. A servo motor is an electric motor with a built in rotation sensor, they are needed for robotics. Say a robot moves its arm by turning a servo motor, the motor would send information concerning the degree of rotation on its axis back to the robot so the robot can keep tabs on the position of its arm, so if something bumps its arm it will know it and so-on.

Electric motors are the commonly used actuator in electromagnetic systems of all types. They are made in variety of configurations and sizes for applications ranging from activating precision movements to powering diesel-electric locomotives. The laboratory motors are small servomotors, which might be used for positioning and speed control applications in a variety of automated machines. They are DC motors. The armature is driven by an external DC

voltage that produces the motor torque and results in the motor speed. The armature current produced by the applied voltage interacts with the permanent magnet field to produce current and motion. The servo DC motor is basically a transducer that converts electric energy into mechanical energy. The torque developed on the motor shaft is directly proportional to the field flux and the armature current. The DC servo motors are very expensive in comparison to ac servo motors because of brushes and commentators. These motors have relatively low torque to volume and torque to inertia ratio; however the characteristics of dc motors are quite linear and are easy to control.

Control motion of the robot manipulator, it is considered one of the most vital and powerful issues in robotics fields because the robot operation must be accurate, without affected surrounding circumstances. Various controllers have been designed and applied in the robot manipulator.PID controller may be the most widely used controller in the industrial and commercial applications for the early decades, due to its simplicity of designing and implementation. Moreover they are available at little costs. One of the drawbacks for using PID control techniques is that, they are not sufficient to obtain the desired tracking control performance because of the nonlinearity of the robot manipulator due to unpredictable environment. Hence, a lot of time is required to tune PID parameters.

The goal of this thesis is to present an engineering approach to control six Degree Of Freedom (6DOF) robot arm. The performance of the controllers will be based on high precision eliminating the overshoot, achieving zero steady state error, damping the unwanted vibration of the robot manipulator, and handling the unpredictable disturbances using PID controller technique.

1.2 problem statement

The mentor robot controller gives different output performance the movement. The performance of a mentor robot arm can be analyzed by

evaluating the movement from an initial position to a final position. And we have some problems exists in this controller such as an external disturbances and nonlinearities. Therefore, it is difficult to eliminate these undesired effects in order to achieve good stability and precise tracking control performance. Sometime the classical linear controller gives deviation in output response from desired input .In this thesis PID controller is used to reduce or minimize the error between actual input and desired output. The PID controller must achieve some specifications such as reducing overshoot, minimizing rising time and eliminating the steady state error.

1.3Objectives

The designing of PID controller of 6 DOF robot arm (mentor) for control system is main aim of this thesis. To achieve this aim the objectives of this research are formulated as flow:

1. To develop the controller for robot (mentor) using classical control system.
2. To compare the performance between the control system and PID control system by the evaluating output response.
3. To design and implement PID controller to the model of robot (mentor) using the MATLAB/Simulink environment and show that in real time response.
4. To achieve more stability and accuracy movement to robotic arm that may be becomes motivation to design other robot in different fields.

1.4 Methodology

The research work was undertaking in the following stages:

1. The main components of system are arduino Uno unit, motor driver unit, power supply unit, and simulink programs.

2. Arduino Uno controller executes the stored program in memory to turn DC motors to reach required point with low cost and low power consumption.
3. The l293 motor driver chip is medium device connected between controller board and tow dc motors.
4. Controller system is responsible for generating rotational movement control signals and commands to each motor driver channel. The signal supplied by controller to the motor driver has a 5v DC.
5. Transfer function of DC motors will be obtained by using system Identification using data driven control in real time mode.
6. Design digital PID controller using matlab&simulink to reduce the overshoot, rise time, and steady state error.
7. Compare between the system before and after PID controller using simulation mode.
8. Implement PID controller on Arduino Uno and compare between desired input angle and actual output angle in real time mode.

Figure (1.1) shows the general system block diagram.

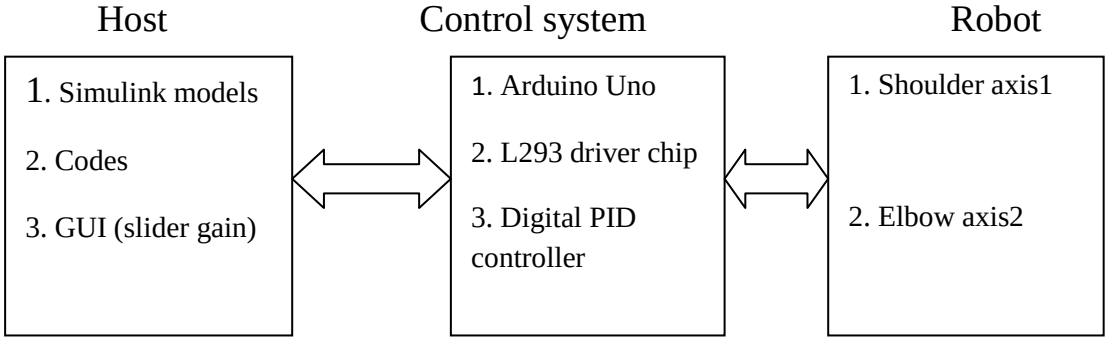


Figure 1.1: General system Block Diagram

Figure (1.2) shows the closed loop motor control system contains: Power supply unit, Host computer, Arduino Uno, L293D, Potentiometer, and DC motor.

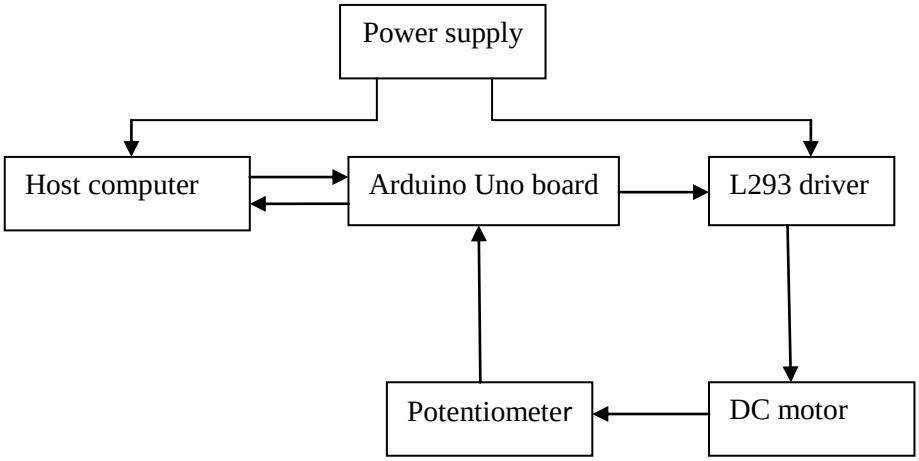


Figure 1.2: closed loop motor control system

1.5 Thesis organization

This thesis is contains from five chapters. Organized as follows: Chapter one shows an introduction of the thesis including thesis importance, problem definition, objectives of the thesis, and applied methodology. Chapter two gives a historical background of PID controller used to arm robotics and describes the principal working of them, then a literature review of previous studies in this field are presented. Chapter three shows a detailed description of the methodology that implemented to design PID controller to robotic arm. Chapter four describes the steps that followed to carry out the experimental and tests and discussed the obtained results. Chapter five concludes this thesis and states the future work and recommendations.

Chapter Two

Background Reading and Literature Review

2.1 Introduction

In recent years, industrial and commercial systems with high efficiency and great Performance have taken advantages of robot technology. Large number of control Researches and numerous control applications were presented during the last years, Concentrated on control of robotic systems. Robot manipulator field is one of the interested fields in industrial, educational and medical applications. It works in Unpredictable, hazard and inhospitable circumstances which human cannot reach. For example, working in chemical or nuclear reactors is very dangerous, while when a robot instead human it involves no risk to human life. Therefore, modeling and analysis of the robot manipulators and applying control techniques are very important before using them in these circumstances to work with high accuracy.

The essential problem is to study the robot manipulator problem from two sides: The first one is the modeling the direct current (DC) motor because it is an important issue in a robot manipulator. The second problem is the design PID controller to the robot manipulator.

The main objective of this thesis is concerned with designing a PID controller for the motion of the robot manipulator to meet the requirement of the desired trajectory input with suitable error and disturbance values. Proportional Integral Derivative (PID) controller may be the most widely used controller in the industrial and commercial applications for the early decades, due to its simplicity of designing and implementation, so the first attempt is to apply PID control; however, PID does not give optimal performance due to the nonlinear elements. Robot manipulators are classified as nonlinear systems, so classical controllers

are not sufficient to give the best results so that we will recommended to design another controller to get the optimal results.

2.2 Robotics

Robotic is branch of engineering and science that include mechanical engineering, electrical engineering, computer science, and others. Robotics deals with design, construction, operation, and use of robots as well as computer system for their control, sensory feedback and information processing.These technologies are used is used to develop machines that can substitute for humans. Robot can be used in any situation and for any purpose, but today many are used in dangerous environment (including bomb detection and deactivation), manufacturing processes, where humans cannot survive. Robots can take any form but some are made to resemble humans in appearance.

The concepts of creating machines that can operate autonomously dates back to classical times, but research into the functionality and potential uses of robot did not grow substantially until the 20^{th} century. Throughout history, it has been frequently assumed that robots will one day be able to mimic human behavior and manage tasks in human-like fashion. Today robotics is rapidly growing field, as technological advances continue; researching, designing and building new robots serve various practical purpose, whether domestically, commercially, or military. The word robotics was derived from word robot, which was introduced to the public by Czech writer Karel Capek in his play R.U.R (ROSSUM'S universal robots), which published in 1920.The word Robot comes from Slavic word robota which mean labor.

2.2.1 History of robots

The history of robots has its origins on ancient world .The modern concept began to be developed with onset of the industrial revolution, which allowed the use of complex mechanics, and the subsequent introduction of electricity .This made it possible to power machines with small compacts motors. In early 20^{th}

century, the notion of a humanoid machine was developed. Today one can envisage human size robot with capacity for near-human thought and movement. Robotics comes from the Russian word “robota” as slave, Joseph E.engelberger is known as the father of robotics. The first uses of modern robots were in factories in industrial robot-simple fixed machines capable of manufacturing tasks which allowed production with less need for human assistance. Digitally controlled industrial robots and robot using artificial intelligence have been built since the 1960s.

The next 100-150 years saw many innovative engineering solutions to pressing problems in industry. A rotary crane equipped with a motorized gripper to remove hot ingots from a furnace was developed by Babbitt in 1892.Pollard invented a mechanical arm for spray painting in 1938.In December 1996, Honda demonstrated the Honda Humanoid, a robot with two legs and two arms that is designed for use in a typical domestic environment. The 210kg prototype has 30 degrees of freedom. It is equipped with cameras, gyroscopes, accelerometers, and force sensors at the wrists and feet. It is able to walk around, climb a flight of stairs, sit down on a chair, stand up from a sitting position and lift payloads of 10lbs.

In 1967, Ralph Moser from General Electric developed a four legged vehicle with funding from the Department of Defense. The vehicle was operated much like the electric tele_operators of the late 1940’s. A human operator would control handles (at the master end) to coordinate the multiple joints at the legs (the slave end). The coordination task was very tedious, and the problems associated with stable walking were never quite resolved. In 1983, Ode tics, Inc., a U.S. company, developed a six legged device that could walk over obstacles while lifting loads up to 2-3 times its weight. While this vehicle was not

teleported (in the sense of Goetz' tele -operators or the GE quadruped), it had to be controlled by a human operator.

2.2.2 Applications of robots

As more and more robots are designed to specific tasks this method of classification becomes more relevant. For example, many robots are designed for assembly work, which may not be readily adaptable for other application. Some robots are designed for heavy load manipulation, and are labelled as heavy duty robots.....etc. Current and potential application includes:

a) Machine loading: where robots supply parts to or remove parts from other machines, that means just handling parts within a set of operations without perform any operation on the part.

b) Pick and place operations: where the robot picks up parts and places them elsewhere.

c) Welding: where the robot along with proper setups and a welding end effectors, is used to weld parts together. This is one of the most common applications of robot in the auto industry.

d) Painting: very common application of robots, especially in the automobile industry.

e) Inspection of parts: circuit's board, and other similar product, is also common applications of robots.

f)_Sampling: with robots is used in many industries, including in agriculture. Sampling can be similar to pick up and place operation and inspection, except that is performed only on a certain number of products.

g) Assembly operations: are among the most difficult for the robot to do. Many of the assembling tasks are complicated as well as the required pushing, turning, pending and snapping the tabs to connect the parts.

h).Manufacturing: by robots may include many different operations, as material removal, drilling, cutting, etc. it also includes insertion of parts.

I). Medical applications: are also becoming increasingly common. For example, the Robotic was designed to assist a surgeon operation, cutting the bones, drilling holes in bodies, etc.

2.2.3 Robotic Arm

Robot must be able to reach a point in space within six axes by moving forward and backward, to the left and right, and up and down. Robot manipulator may be classified according to the type of movement needed to complete the task. The jointed arm robot closely resembles the human arm. It is a combination of cylindrical and articulated configurations, the arm of the robot is connected to the base with a twisting joint, and the links in the arm are connected by rotary joints. Many commercially available robots have this configuration.

This structure is very flexible and has the ability to reach over obstructions. It can generally achieve any position and orientation within the working envelope in eight different ways. All joints can be sealed from the environment. When driving these robots in their natural co-ordinate system (joint space) the motion of the robot from one point to another can be difficult to visualize as the robot will move each joint through the minimum angle required. This means that the motion of the tool will not be a straight line. This structure of robots is used for a wide range of applications including paint spraying; arc and spot welding, machine tending, fettling, assembly operations and handling at die casting or fettling machines.

2.2.4Mentor desktop robots

Figure 2.1 show the robotic arm (mentor) its low priced, versatile, robust and reliable, the mentor is ideal entry point into the world of robotics and computer integrated manufacturing. The mentor has articulated arm with joints similar to that of human arm and this configuration is widely used industrially each of the axes is driven by a DC servo motor with its position monitored by a potentiometer. A built-in controller provides close- loop control of the system

and constantly provides monitoring data from computer. Programming from the computer by setting the data for each axis. Specification of mentor arm

- Axis (waist) 0: Angular movement 210°.Axel centre from top of base 185mm.
- Axis (shoulder) 1: angular movement 180°.arm length between axel centers 165mm.
- Axis (elbow) 2: angular movement 320°.Arm length between axels 150mm.
- Axis (left wrist axel) 3: Angular movement 320°.
- Axis (right wrist axel) 4: Angular movement 230°.
- Wrist pitch: Angular movement140°, wrist rolls: movement 320°, gripper: jaw opening 45mm. jaw pressure 10N.
- Repeatability: 2mm, lifting: 1000gm at full reach,: 428mm from axes 1axel centre.
- Base: 320×270×189mm, control system: 8bit (0.4%), work cell: digital output8.
- Interface: Universal Serial Bus(USB) connector to host Personal Computer (PC). Digital input8.analouge input4.

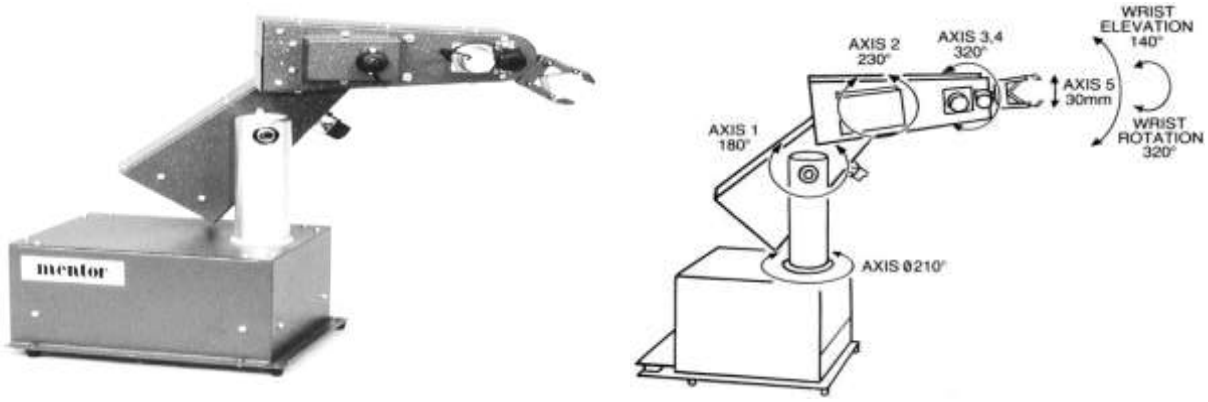
The main features of mentor arm are:

- Five axes human arm configuration, plus gripper.
- Built-in control system.
- Ready to run using hand controlled simulator.
- Easy to program using WALLI software.
- Robust metal construction with an arm reaches in excess of 400mm.
- Can lift 1Kg at full reach.
- Maximum base to gripper height 780mm fully extended.

Table 2.1 shows the properties of mentor robot arm.

Table 2.1: Mentor arm robot properties

Properties	Specification
Repeatability	2 degree
Accuracy	1.5 degree
Lifting (payload)	1000gm at full reach
Gripper	Jaw opening 45mm
Base	320x270x189 (mm)



(A) (B)

Figure 2.1: (A) Mentor desktop robotic, (B) Maximum allowable movement

Table 2.2 shows the structure and movement of mentor robot arm.

Table (2.2): Movement range of basic structure

Structure	Movement type	Maximum allowable
Waist	Rotation	(Θ1) $0<\Theta1 \leq 360°$
Arm	Rotation	(Θ2) $0<\Theta2 \leq 180°$
Forearm	Rotation	(Θ3) $0<\Theta3 \leq 200°$
Wrist	Rotation	(Θ4) $0<\Theta4 \leq 360°$
Shoulder	Linear	(L1) $0<L1 \leq 8$ inch
Base	Linear	(L2) $0<L2 \leq 12$ inch

2.3 Direct Current motor

Direct current motors have variable characteristics and are used extensively in variable-speed drives. DC motors can provide a high starting torque and it is also possible to obtain speed control over a wide range. The methods of speed control

are normally simpler and less expensive than those of ac drives [1].Due to its wide range of application different functional types of dc motor are available in the market for specific requirements. Figure 2.1 show the types of DC motor.

Figure 2.2: Types of DC motor

Direct current motors hold the very important status in the electric driving automatic control system. Relative to the alternating current motor, the performance of direct current motor's speed control is much better. It is the first choice in the applications which require wide range of speed regulation and high-precision speed, and it has been widely used in computerized numerical control machine tools and process control [2]. DC motors can be used in various applications and can be used as various sizes and rates. Today their uses isn't limited in the car applications (electrics vehicle), in applications of weak power using battery system (motor of toy) or for the electric traction in the multi-machine systems too. The speed of DC motor can be adjusted to a great extent as to provide controllability easy and high performance [3].

The speed of DC motors changes with the load torque. To maintain a constant speed, the armature (and or field) voltage should be varied continuously by varying the delay angle of AC-DC converters or duty cycle of DC-DC converters. In practical systems it is required to operate the drive at constant torque or constant power; in addition controlled acceleration and deceleration are required. Nowadays, most industrial drives operate as closed-loop feedback system due to the system has the advantages of improves accuracy, fast dynamic

response and reduced effect of load disturbances and system nonlinearities. Figure 2.3shows the block diagram of DC motor closed loop control.

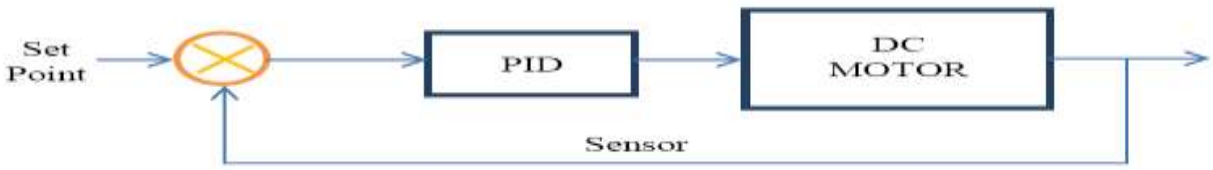


Figure 2.3: The block diagram of the DC motor closed-loop control

As it is seen from figure 2.2 the block diagram of the DC motor closed-loop control, the speed sensor (encoder) measure the speed of the DC motor. In these loops we have the actual speed of the DC motor with the desired one. The DC speed measurement gives the actual speed value. The error between theoretical and practical values is corrected with PID controller. The parameters of the PID controller are determined with MATLAB results which will be explained in the following sections. The output of the PID controller gives the duty cycle of the square wave generator [4].

2.4 Proportional integral derivative controller

PID controller is considered the most widely used control technique in control applications. A high number of applications and control engineers had used the PID controller in daily life. PID control offers an easy method of controlling process by varying its parameters. It consists from Proportional (P) controller, Proportional-Integral (PI) controller, Proportional-Derivative (PD) controller and Proportional-Integral-Derivative (PID) controller

2.4.1Proportional (P) controller:

Figure 2.4: Proportional controller block diagram

Proportional (P) controller is mostly used in first order processes with single energy storage to stabilize the unstable process. The main usage of the P controller is to decrease the steady state error of the system. As the proportional gain factor K increases, the steady state error of the system decreases. However, despite the reduction, P control can never manage to eliminate the steady state error of the system. As we increase the proportional gain, it provides smaller amplitude and phase margin, faster dynamics satisfying wider frequency band and larger sensitivity to the noise. Controller can use this only when our system is tolerable to a constant steady state error. In addition, it can be easily concluded that applying P controller decreases the rise time and after a certain value of reduction on the steady state error, increasing K only leads to overshoot of the system response. P control also causes oscillation if sufficiently aggressive in the presence of lags and/or dead time. The more lags (higher order), the more problem it leads. Plus, it directly amplifies process noise [4].

2.4.2Proportional-Derivative (PD) controller

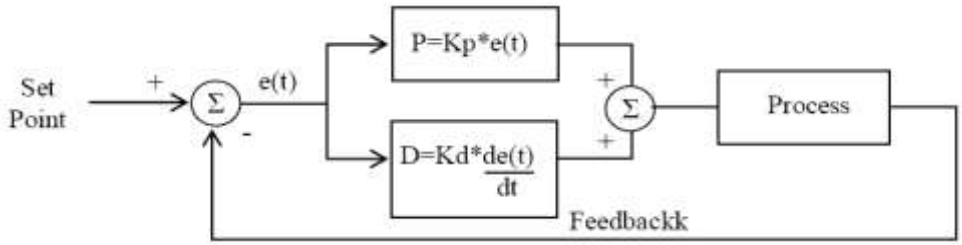


Figure 2.5: Proportional-Derivative controller block diagram

The aim of using Proportional-Derivative (PD) controller is to increase the stability of the system by improving control since it has an ability to predict the future error of the system response. In order to avoid effects of the sudden change in the value of the error signal, the derivative is taken from the output response of the system variable instead of the error signal. Therefore, D mode is designed to be proportional to the change of the output variable to prevent the sudden changes occurring in the control output resulting from sudden changes in

the error signal. In addition D directly amplifies process noise therefore D-only control is not used [4].

2.4.3Proportional-Integral-Derivative (PID) controller

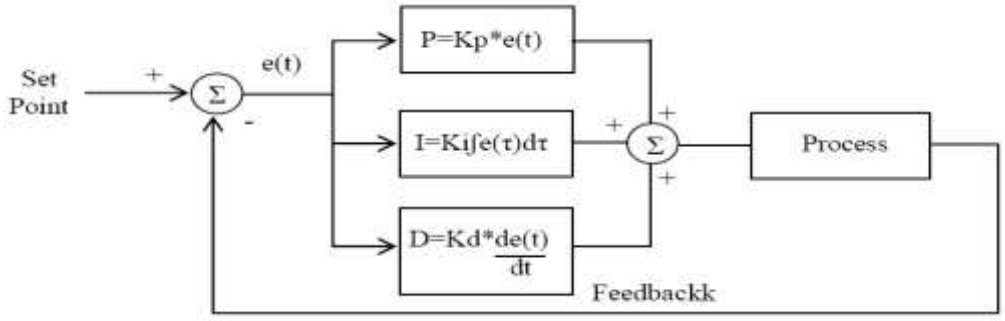


Figure 2.6: Proportional-integral-derivative controller block diagram

Proportional-Integral-Derivative (PID) controller has the optimum control dynamics including zero steady state error, fast response (short rise time), no oscillations and higher stability. The necessity of using a derivative gain component in addition to the PI controller is to eliminate the overshoot and the oscillations occurring in the output response of the system. One of the main advantages of the PID controller is that it can be used with higher order processes including more than single energy storage [4].

PID controllers are widely used in industrial practice over 60 years ago. The invention of PID control is in 1910 (largely owing to Elmer Sperry's ship autopilot) and the straightforward Ziegler-Nichols (Z-N) tuning rule in 1942. Today, PID is used in more than 90% of practical control systems, ranging from consumer electronics such as cameras to industrial processes such as chemical processes. The PID controller helps get our output (velocity, temperature, position) where we want it, in a short time, with minimal overshoot, and with little error. It also the most adopted controllers in the industry due to the good cost and given benefits to the industry. Many nonlinear processes can be controlled using the well-known and industrially proven PID controller [5].

The control structure used was the PID as a base for software development. This type of controller was initially chosen, because it is generally applied to most of the control systems of continuous processes, proving its usefulness by providing a generally satisfactory control. The connection among the proportional, integral and derivative actions result in the PID controller, which can be found in various formats. Among the existing formats, the one chosen to be implemented in the experiment was parallel PID, since it is the most common among industrial controllers. Figure 6 shows the block diagram of a closed loop control system using PID controller [6].Mathematical description of PID controller is:

$$U(t) = Kpe(t) + Ki\int_0^t Kd\frac{de(t)}{dt} \qquad (2.1)$$

Where

t = Time

U (t) = Output

e (t) = Error = set point – process variable

K_p = Proportional controller mode gain

K_i = Integral controller mode gain

K_d = Derivative controller mode gain

Using above equation of the PID controller in time domain it is possible to develop an equation of a digital PID controller in order to use it on an arduino.

2.4.4**Characteristic of P, I and D Controllers**

The function of proportional controller (K_p) will have the effect of reducing the rise time and will reduce, but never eliminate, the steady-state error. An integral control (K_i) will have the effect of eliminating the steady-state error, but it may make the transient response worse. A derivative control (K_d) will have the effect of increasing the stability of the system, reducing the overshoot, and

improving the transient response. Effects of each of controllers K_p, K_d, and K_i on a closed-loop system are summarized in the Table3.2.

Table 2.3: Characteristic of P, I and D controllers

Response	Rise Time	Overshoot	Settling Time	Steady State Error
K_P	Decrease	Increase	Small Change	Decrease
K_i	Decrease	Increase	Increase	Eliminate
K_d	Small Change	Decrease	Decrease	Small Change

2.4.5 Tuning PID methods

The meaning of tuning is adjustment of control parameters to the optimum values for the desired control response. The stability is a basic requirement. However, different systems have different behavior, different applications have different requirements, and requirements may conflict with one another. PID tuning is a difficult problem, even though there are only three parameters and in principle is simple to describe, because it must satisfy complex criteria within the limitations of PID control. There are accordingly various methods for loop tuning, some of them:

(i) Manual tuning method

Using manual tuning method, parameters are adjusted by watching system responses. K_p, K_i, K_d are changed until desired or required system response is obtained. Even though this method is simple, it should be used by experienced personal. For example, using one manual tuning method, firstly, K_i and K_d are set to zero. Then, the K_p is increased until the output of the loop oscillates, after obtaining optimum K_p value, it should be set to approximately half of that value for a "quarter amplitude decay" type response. Then, K_i is increased until any offset is corrected in sufficient time for the process. However, too much K_i will

cause instability. Finally, K_d is increased, until the loop is acceptably quick to reach its reference after a load disturbance. However, too much K_d also will cause excessive response and overshoot. A fast PID loop tuning usually overshoots slightly to reach the set point more quickly; however, some systems cannot accept overshoot, in which case an over-damped closed-loop system is required, which will require a K_p setting Significantly less than half that of the K_p setting causing oscillation. Referring to table 2.4, the effects of changing control parameters can be seen.

Table 2.4: Effect of changing control parameter

Response	Rise Time	Overshoot	Settling Time	Steady State Error	Stability
K_P	Decrease	Increase	Small Change	Decrease	Worse
K_i	Decrease	Increase	Increase	Eliminate	Worse
K_d	Minor Change	Minor Change	Minor Change	No Change	If K_d small better

(ii) Ziegler–Nichols tuning method

This method was introduced by John G. Ziegler and Nathaniel B. Nichols in the 1940s. The Ziegler-Nichols' closed loop method is based on experiments executed on an established control loop (a real system or a simulated system). The tuning procedure is as follows:

a) Bring the process to (or as close to as possible) the specified operating point of the control system to ensure that the controller during the tuning is "feeling" representative process dynamic and to minimize the chance that variables during the tuning reach limits. The process is brought to the operating point by manually adjusting the control variable, with the controller in manual mode, until the process variable is approximately equal to the set-point.

b) Tune the PID controller into a P controller by setting set $T_i = \infty$ and $T_d = 0$. Initially, gain K_p is set to "0". Close the control loop by setting the controller in automatic mode.

c) Increase K_p value until there are sustained oscillations in the signals in the control system, e.g. in the process measurement, after an excitation of the system. (The sustained oscillations correspond to the system being on the stability limit.) This K_p value is denoted the ultimate (or critical) gain, Kpu. The excitation can be a step in the set-point. This step must be small, for example 5% of the maximum set-point range, so that the process is not driven too far away from the operating point where the dynamic properties of the process may be different. On the other hand, the step must not be too small, or it may be difficult to observe the oscillations due to the inevitable measurement noise. This is important that Kpu is found without the control signal being driven to any saturation limit (maximum or minimum value) during the oscillations. If such limits are reached, there will be sustained oscillations for any (large) value of K_p, e.g. 1000000, and the resulting K_p value is useless (the control system will probably be unstable). One way to say this is that Kpu must be the smallest K_p value that drives the control loop into sustained oscillations.

d) Measure the ultimate (or critical) period Pu of the sustained oscillations.

e) Calculate the controller parameter values according to Table 2.5, and these parameter values are used in the controller. If the stability of the control loop is poor, stability is improved by decreasing K_p, for example a 20% decrease.

Table 2.5: Ziegler-Nichols tuning method, gain parameters calculation

Type of Controller	K_P	T_i	T_d
P	$0.5K_{cr}$	∞	0
PI	$0.45K_{cr}$	$0.83P_{cr}$	0
PID	$0.6K_{cr}$	$0.5P_{cr}$	$0.125P_{cr}$

(iii) PID Tuning Software Method

There is some prepared software that they can easily calculate the gain parameter. Any kind of theoretical methods can be selected in some these methods [5]. Here the examples of software most used:

(1) MATLAB /SIMULINK PID controller tuning,

(2) BESTune,

(3) Exper tune etc.

Table 2.6 advantages and disadvantages of tuning methods.

Method	Advantages	Disadvantage
Manual Tuning	No math required. Online method.	Requires experienced personnel.
Ziegler- Nichols	Proven method. Online Method	Process upset, Some Trial and error, very aggressive tunning.
Software Tool	Consistent tuning. Online or offline method. May include value and sensor analysis. Allow simulation before downloading. Can support non-steady state tuning.	

2.5 Arduino controller

There are many types of arduino controller as flow:

a) Arduino Uno.

b) Arduino mega.

c) Arduino mini.

d) Arduino Nano.

e) Arduino mini pro.

f) Arduino BT.

g) Arduino Leonardo.

In this thesis we will discuss type (a) that we will use in section of system identification to DC motors, and type (b) that we will use in controller design of robotic arm.

a) Arduino Uno controller

As seen in figure 2.7, CPU module interacts with all modules; CPU is based on Arduino Uno board. Arduino is an open hardware/software platform, The Arduino Uno is a microcontroller board based on the ATmega328. Arduino becomes a very fast development platform even for those how aren't into electronics' world. Arduino programming becomes much faster because the graphical open-source environment makes it easy to write code and upload it to the board [7].

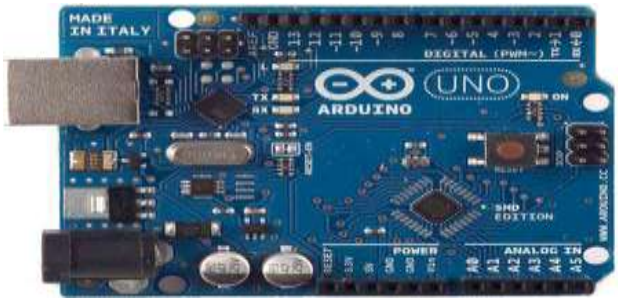


Figure 2.7: Arduino Uno Board

Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. Arduino can sense the environment by receiving input from a variety of sensors and can affect its surroundings. Arduino projects can be stand-alone, or they can communicate with software running on a computer. In this development, Arduino UNO is used as the main controller because it satisfies these conditions as Table 2.5 [8]:

Table 2.7: Specification of Arduino Uno Board

Microcontroller	ATmega 328
Operating system	5V
Input voltage (recommended)	7-12V
Input voltage(limits)	6-20V
Digital I/O pins	14(of which 6 provide PWM output)
Analog input pins	6
DC current per I/O pin	40 mA

DC current for 3.3V pin	50 mA
Flash memory	32KB(ATmega328) of which 0.5KB used by boot loader
EEPROM	1KB(ATmega328)
SRAM	2KB(ATmega 328)
Clock speed	16MHz

2.6MATLAB Software

MATLAB software can be used for a range of applications, including signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology. A million of engineers and scientists in industry and academia use MATLAB as the language of technical computing. A new control of a drive system is formulated and it is often convenient to study the system performance by using MATLAB/ SIMULINK support package.

2.7 Literature review

The work presented by Mohammed Hussein [14] controlling a robotic arm is a challenging issue from the past days. In this thesis a controller was developed and built inside six axes robotic arm. The three main units in the designed controller were MEGA 2560 Arduino microcontroller unit, driver unit, and power supply unit. In the Driver unit L293D H-bridge was used. Six DC motors 12VDC, with related potentiometers for positional feedback signal, were used for controlling the movements of the joints in different directions. The six servo DC motors used in this system were DME 33 series with gearbox type 5C from NIDEC Servo Corporation. Mentor robotic arm was used for testing this controller. Proteous software was used for simulating the movements of the six motors. The controller was connected to a personal computer through its USB port. A graphical user interface window was designed in the personal computer based on Labview software. Commands for joints movements were sent using an ordinary game joystick. This system proposed low cost, 300$, low power

consumption, 8 VADC, small size, and high accuracy, 0.5 degree, controller solution for complex robotic arm.

Position control performed using independent joint control in [9]. This method was using PID controller and it worked by controlling each joint independently? The coupling effect between the joints and links could be ignored if the gear ratio was large.

Industrial robot manipulators are generally used in positioning and handling devices. Rapid technology changes in industrial robot manipulator urged human to develop and improve the performance of control strategy which can be used to replace human in hazardous, complex and boring tasks [10]. Generally, the dynamics behavior of robot manipulators can be described by the nonlinearity and external input parameter such as in friction, load and disturbances [11].

Nowadays, PID controller is a good control technique in industrial automation applications due to its simplicity and robustness [12]. Although PID is most widely used controller in industry, it has several drawbacks. One of the drawbacks is that the PID controller is not sufficient to obtain the desired tracking control performance because of the nonlinearity of the robot manipulator [13]. Therefore, PID controller is not a good technique to control a system with nonlinear environment.

An approach to combine a fuzzy logic controller and PID controller was presented by [4]. In that paper, he replaced the proportional term in the PID with fuzzy P controller and thus became the fuzzy P+ID controller.

Chapter three

Design and Implementation of PID Controller

3.1 Introduction

This chapter show that how to estimate the transfer function of DC motors, how to design PID controller for tow axis of robot (mentor), how to implement it to the system, and how to operate the system in real time mode. The system can divide into groups: hardware implementation (DC motors on robot body, arduino Uno board, L293D Driver, host computer, and power supply units), software implementation (MATLAB&SIMULINK, system identification software, and PID controller implantation software).

3.2Hardware Implementation

The main hardware implementation as follows:

3.2.1 DC motor

DC motor is common actuator found in any mechanical systems and industrial applications such as industrial and educational robot .the motor has rotary movement and when combined with mechanical part it can provide translation movement for desired link. Generally the function of DC motor is to convert the electrical energy to mechanical energy. The DC motor using in this system shows in Figure 3.1. Its Japanese product item#DME33S5C500A, DME series motors with gearbox5C [22]. Table 3.1 shows the DC motor specifications.

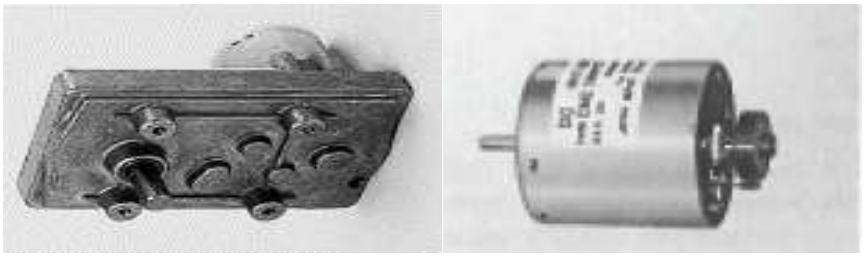


Figure 3.1: The DC Motor with Gearbox (DME33 series)

Table 3-1: Used DC Motor Specifications

Product Type	DC Motor with Gearbox 5C
Series	DME33
Model Code	SA
Gear Ratio	500
Rated Voltage	12 VDC
Rated Output	3.0 W
Current Rating	420 Ma
Rated Speed	9 r/min
Rated torque	68.05 oz-in 0.48N.m
Diameter	1.26 in 32mm
Frame size	94.5*50mm 3.72*2.7262in

To estimate the transfer function of (DME33S5C500A) DC motors that using in 6DOF robot arm (mentor), distributed on five positions showed in figure (2.1) and structure and movement showed in table (2.2) we will use the system identification process that is depend on DDC approach in step(1)and step(2). In this thesis we will estimate tow transfer functions for DC motor positioned in tow different location with different loads. Data driven control (DDC) is approach requires measured input-output data and used to design controller when plant model is not available. DDC workflow is shown in Figure 3.2.

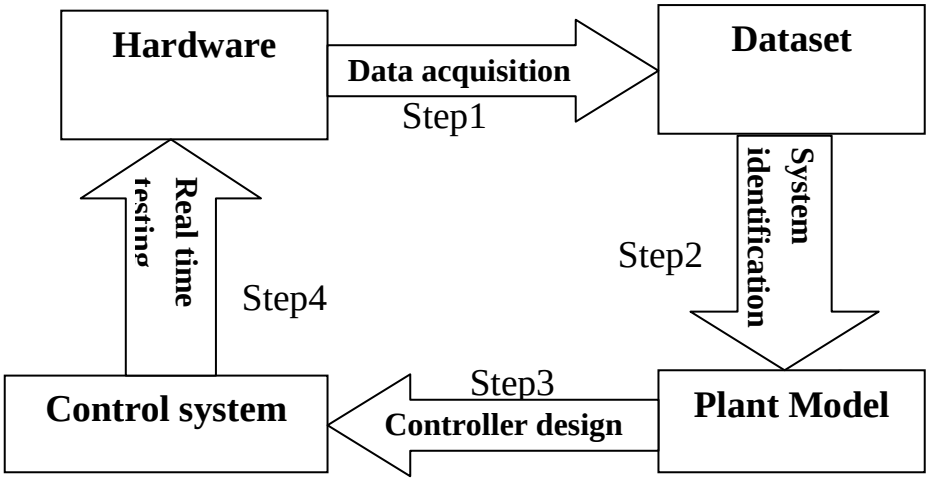


Figure 3.2: DDC block diagram

The workflow of DDC shows the steps of data driven control have four steps as follow:

a) Data acquisition:

To create data acquisition system there are two steps must be considered: step1 the hardware setup and step2 run on target hardware. The hardware step in this thesis composed from DC motors (DME33S5C500A) connected to arduino Uno board throw L293 DC motor driver, the driver ICS is basically used to increase the current capability in order to able to drive the motor. According to this the design of feedback controller for DC motor (DME33S5C500A) to tracking reference motor position. The arduino Uno receive the reference motor position (input) from host computer and generate the signal and send it to the motor throw DC motor driver to turn the motor shaft, then potentiometer is used to measure the angle of motor rotation and send back the signal again to arduino board throw, finally arduino send the potentiometer signal (output). Figure 3.3show the hardware setup.

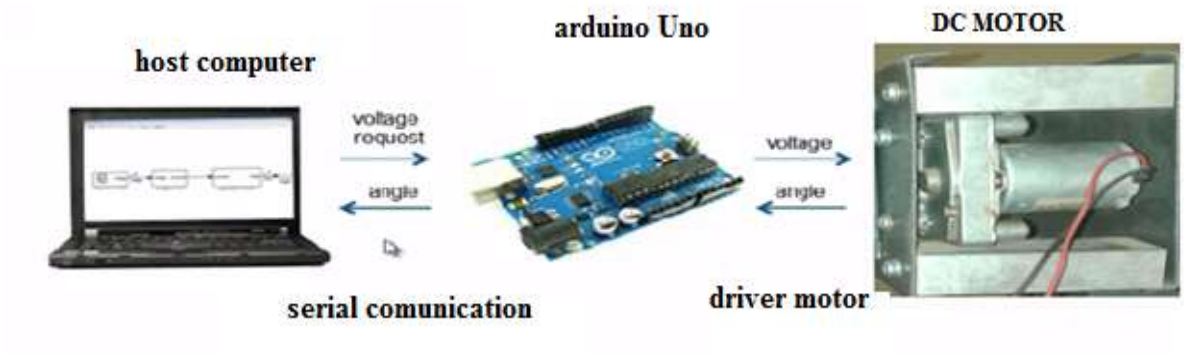


Figure 3.3: hardware setup

Run on target hardware creates an executable file from model, and run it on target hardware .is available from the model tools manual: tool>run target hardware, the result of this procedures is to build data acquisition on arduino Uno board, this model shown in Figure3.3 In software implementation part, the main functions of this model are:

a) Receive reference voltage from host over the serial port.

b) Send voltage request to the motor.

c) Receive angle signal from the motor

d) Send angle signal to host over serial port.

the main functions of this host computer are:

a) Transmit the voltage request to arduino Uno board over serial port.

b) Receive the angle signal from arduino Uno board over serial port.

b) System identification

By performing identification we can get the relationship between model validity and cost to estimate the model Figure 3.4 shown in .

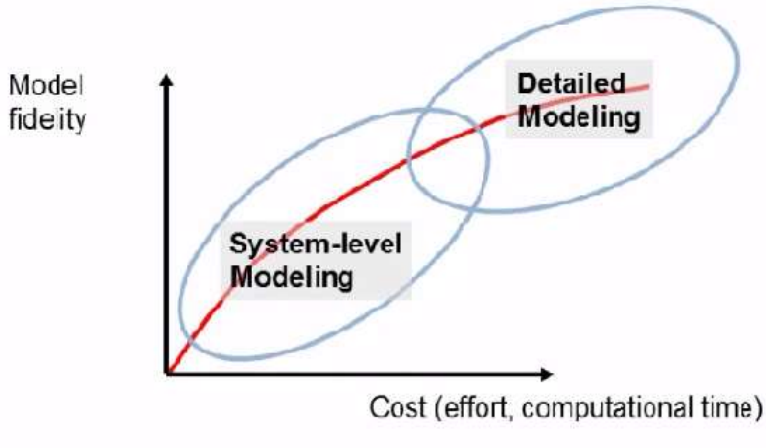


Figure 3.4: model fidelity vs cost

By using system identification toolbox we can model the dynamic that matter for our analysis, also balance cost and model fidelity. The main object of this procedure is to estimate the transfer function of DC motor, the detail of system identification steps and results explained in chapter four.

3.2.2Arduino Uno board

The Arduino Uno is a microcontroller board based on the ATmega 328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16MHz ceramic resonator, a USB connection, a power jack, an ICSP

header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. More information about it in chapter2.

Arduino Uno board in this thesis used in the part of estimating transfer function of DC motor as the part of data driven control system, the main function of arduino UNO here is used as data acquisition system by uploading the model of simulink shown in figure 3.5 on arduino UNO board to be able to receive and transmit signal over serial port.

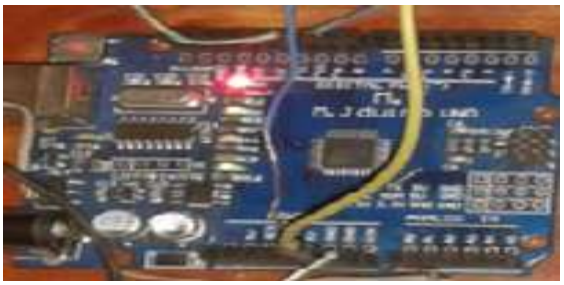


Figure 3.5: arduino UNO board

3.2.3 L293D driver

The arduino port are not powerful enough to drive dc motor directly so we need some kind of driver a very useful and save is to use L293D chip, It can actually control two motors independently. We are just using half the chip in this project. The L293D Driver is a monolithic integrated high voltage, high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as DC and stepping motors) and switching power transistors.

To simplify use as two bridges each pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a lower voltage and internal clamp diodes are included. This device is suitable for use in switching applications at frequencies up to 5 kHz. The L293D is assembled in a 16 lead plastic package which has 4 center pins connected

together and used for heat sinking, Figure 3.6and 3.7 show the block diagram and pin connections.

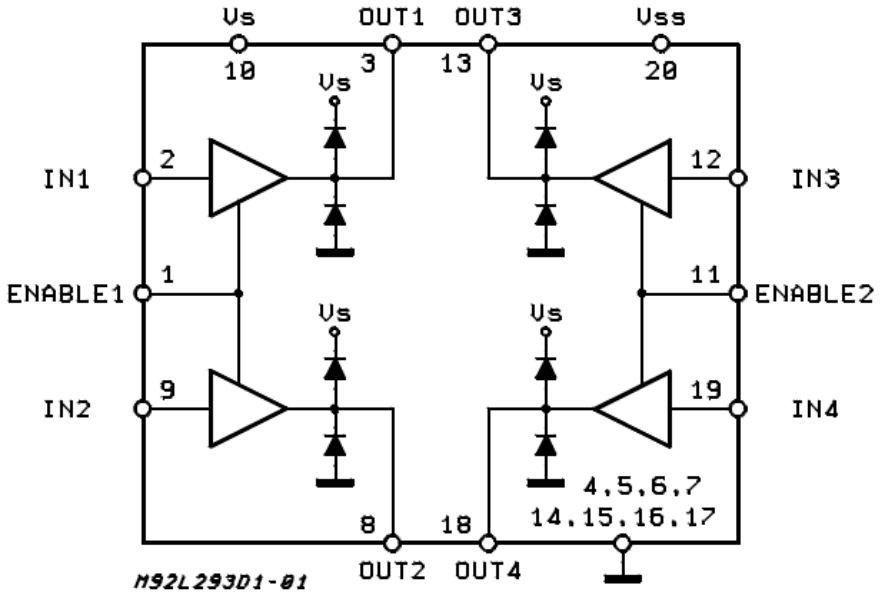


Figure 3.6: schematic diagram of L293D driver

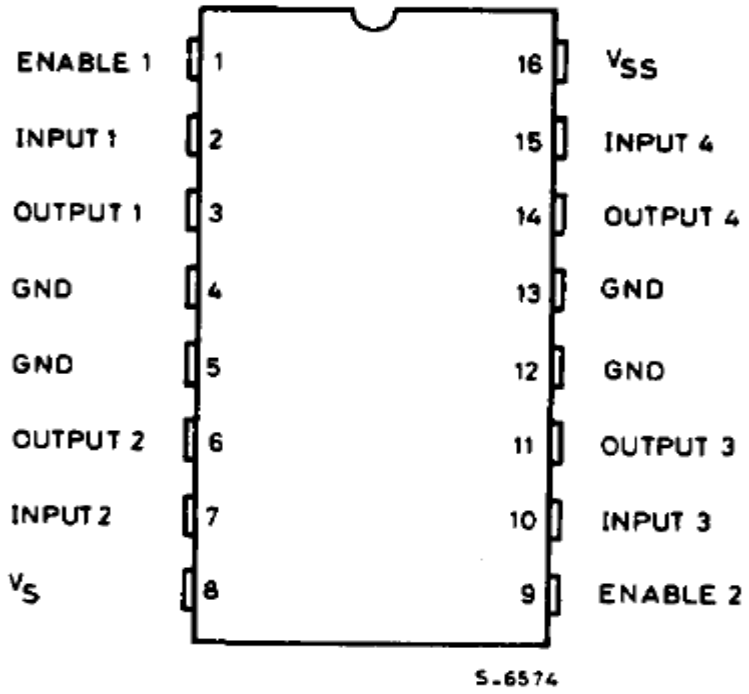


Figure 3.7: L293D pins connections.

Table 3.2 show absolute maximum ratings Absolute Maximum Rating

Table 3.2 absolute maximum ratings of L293D

Symbol	Parameter	Value	Unit
Vs	Supply voltage	36	V
Vss	Logic Supply Voltage	36	V
Vi	Input Voltage	7	V
Ven	Enable Voltage	7	V
Io	Peak Output Current (100 ms non repetitive)	1.2	A
Ptot	Total Power Dissipation at Tpins = 90 °C	4	W
Tstg,Tj	Storage and Junction Temperature	-40 to 150	C

The truth table of L293D is shown in table 3.3

Table (3.3): The truth table of L293D

Enable	Input1	Input2	Function
H	H	L	Turn right
H	L	H	Turn lift
H	L/H	L/H	Stop
L	Either	Either	Stop

This chip is designed to control two DC motors .there are two input and two output for each motor .The connection is shown in Figure 3.8.

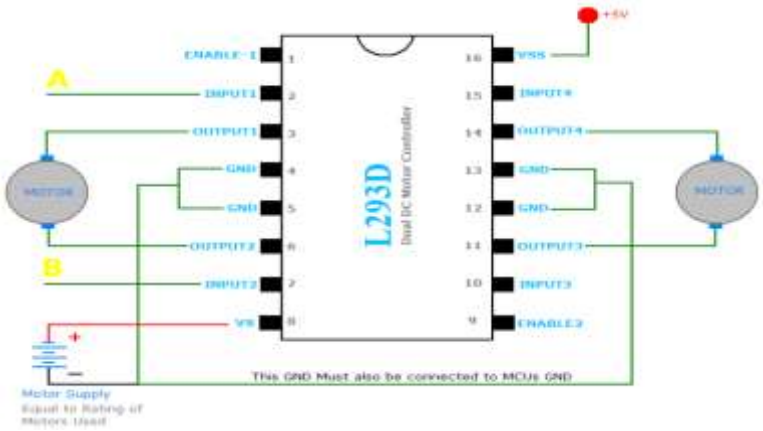


Figure 3.8:Power supply to L293D and DC motors connection

3.3Software Implementation

All software used in this thesis depends on: MATLAB/SIMULINK program. The software divided into parts:

a) System identification software.

b) PID controller design software.

3.3.1MATLAB/SIMULINK

MATLAB (short for Matrix Laboratory) is a special-purpose computer program optimized to perform engineering and scientific calculations; The MATLAB program implements the MATLAB programming language, and provides a very extensive library of predefined functions to make technical programming tasks easier and more efficient. MATLAB is a huge program with an incredibly rich variety of functions. Even the basic version of MATLAB without any toolkits is much richer than other technical programming languages. There are more than 1000 functions in the basic MATLAB product alone, and the toolkits extend this capability with many more functions in various specialties.

(1)Advantages of MATLAB

MATLAB has many advantages compared to conventional computer languages for technical problem solving. Among them are.

a) Ease of use

MATLAB is an interpreted language; it is very easy to use. The program can be used as a scratch pad to evaluate expressions typed at the command line, or it can be used to execute large prewritten programs. Programs may be easily written and modified with the built-in integrated development environment and can be debugged with the MATLAB debugger. Because the language is so easy

to use, it is ideal for the rapid prototyping of new programs. Many program development tools are provided to make the program easy to use. They include an integrated editor/debugger.

b) Platform independence

MATLAB is supported on many different computer systems, providing a large measure of platform independence. At the time of this writing, the language is supported on Windows XP/Vista, Linux, several versions of Unix, and the Macintosh. Programs written on any platform will run on all of the other platforms, and data files written on any platform may be read transparently on any other platform. As a result, programs written in MATLAB can migrate to new platforms when the needs of the user change.

c) Predefined functions

MATLAB comes complete with an extensive library of predefined functions that provide tested and prepackaged solutions to many basic technical tasks. For example, suppose that you are writing a program that must calculate the statistics associated with an input data set. In most languages, you would need to write your own subroutines or functions to implement calculations such as the arithmetic mean, standard deviation, median, and so forth. These and hundreds of other functions are built into the MATLAB language, making your job much easier. In addition to the large library of functions built into the basic MATLAB language, there are many special-purpose toolboxes available to help solve complex problems in specific areas. For example, a user can buy standard toolboxes to solve problems in signal processing, control systems, communications, image processing, and neural networks, among many others. There is also an extensive collection of free user-contributed MATLAB programs that are shared through the MATLAB Web site.

d) Device-independent plotting

Unlike most other computer languages, MATLAB has many integral plotting and imaging commands. The plots and images can be displayed on any graphical output device supported by the computer on which MATLAB is running. This capability makes MATLAB an outstanding tool for visualizing technical data.

e) Graphical user interface

MATLAB includes tools that allow a programmer to interactively construct a graphical user interface (GUI) for his or her program. With this capability, the programmer can design sophisticated data analysis programs that can be operated by relatively inexperienced users.

(2)Disadvantages of MATLAB

MATLAB is an interpreted language .The main disadvantage of interpreted languages is execution speed .when a language is compiled all of the code is analyzed and processed efficiently, before the programmer distributes the application. With the interpreted language, the computer running the program has to analyze and interpret the code (through the interpreter) before it can execute (each and every time), resulting in slower processing performance.

(3)SIMULINK

Simulation link, an add-on product to MATLAB, provides an interactive, graphical environment for modeling, simulating, and analyzing of dynamic systems. It enables rapid construction of virtual prototypes to explore design concepts at any level of detail with minimal effort. For modeling, Simulink provides GUI for building models as block diagrams. It includes a comprehensive library of pre-defined blocks to be used to construct graphical

models of systems using drag-and-drop mouse operations. It supports linear and nonlinear systems, modeled in continuous-time, sampled time, or hybrid of the two. Since students learn efficiently with frequent feedback, the interactive nature of Simulink encourages you to try things out, you can change parameters "on the fly" and immediately see what happens, for "what if" exploration. Lastly, and not the least, Simulink is integrated with MATLAB and data can be easily shared between the programs.

3.3.2System identification software

The main objective of use system identification in this thesis is estimate transfer function of DC motors. There is tow simulinks used in system identification procedure, the first simulink code show in Figure 3.9, and second simulink code show in Figure 3.10 designed by arduino support package.First simulink code uploaded on arduino card to send the input voltage of reference motor position from host computer a and sent it to motor, and receive output voltage (angle) from potentiometer sensor, the second simulink code existed in host computer used to transmit input voltage (reference motor position) to arduino board, and receive the output of dc motor from arduino board throw serial communication board. The full steps of system identification experience and results explained in chapter four.

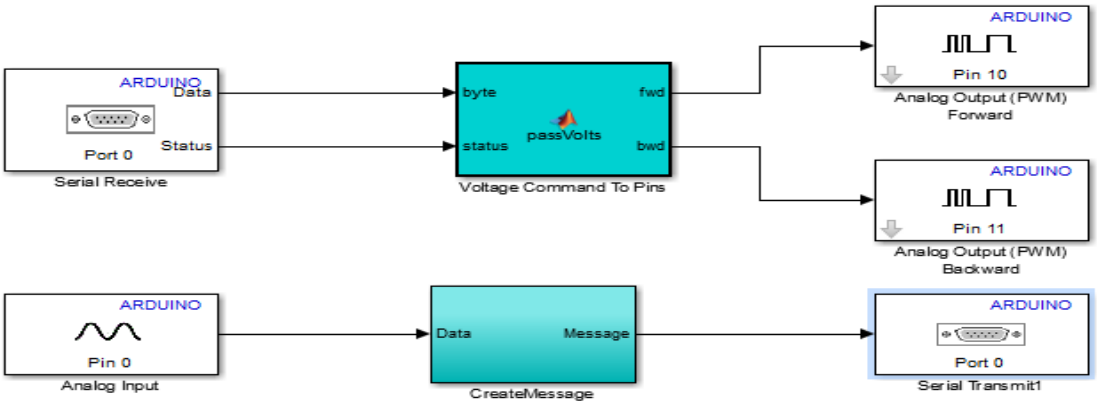


Figure 3.9: Simulink model of data acquisition

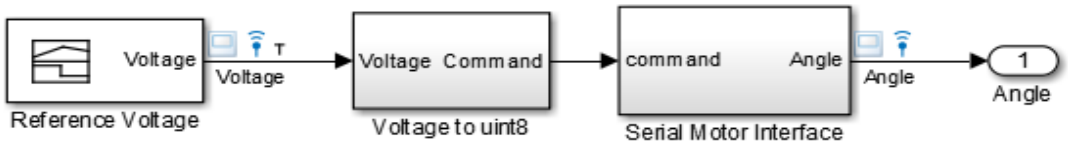


Figure (3.10) Simulink model of host computer

The main parts in Figure 3.9 are:

- Serial receiver: this block shows the port number in arduino used to receive voltage reference signal from host computer.
- Voltage command to pin: this matlab function block will read data coming over serial board routs voltage to do analog output pins on the board.
- Analog input: analog input pin 0 receives the motor position data.
- Create message: in this subsystem complete serial message spent and transmitted back to the host.
- Analog output (pwm) forward: this block use pin10 in analog port to drive the motor in forward motion.
- Analog output (pwm) backward: this block use pin11 in analog port to drive motor in backward motion.
- Serial transmit: this block shows the port number on arduino used to transmit angel signal of motor to host computer.

The main parts in Figure 3.10 are:

1. Reference voltage: reference voltage block here is basically signal build by the create different command signal for example:

   A) Bounds1 command signal as show in Figure 3.11.

Figure 3.11: signal builder bounds1 command signal

b) Bounds5 command signal as shown in Figure3.12.

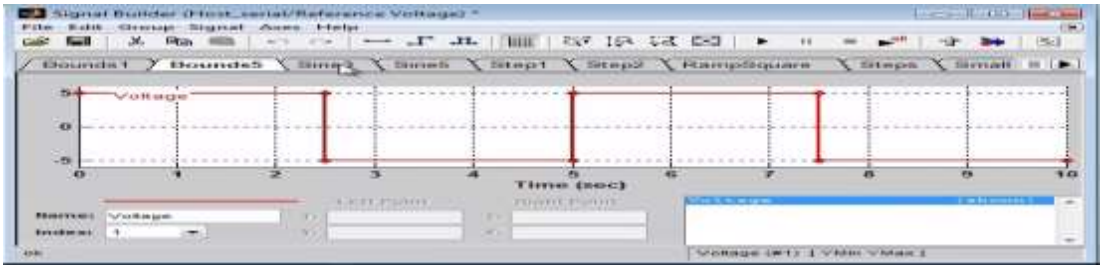


Figure 3.12: signal builder bounds5 command signal

c) Sine 5 command signal as show in Figure 3.13.

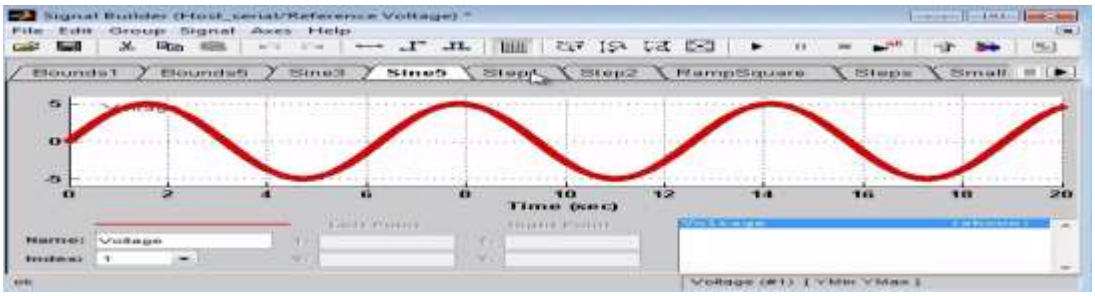


Figure 3.13: signal builder sine 5 command signal

d) Small step command signal as shown in Figure 3.14.

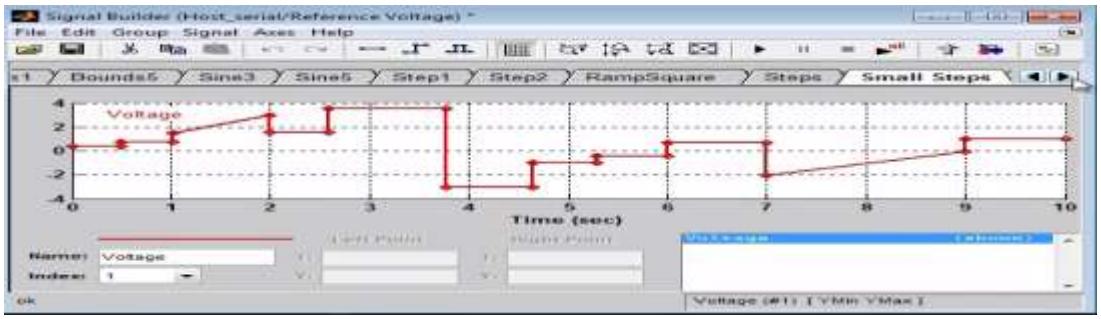


Figure (3.14) signal builder small step command signal

2. Serial motor interface:

This block uses to communicate with motor and it is consist of matlab function block called serial interface to arduino used to send and receive data and data receive changed to angle by 10-bit angle block subsystem.

3) Voltage to unit8**:** used to preprocess to serial motor interface over serial port.

3.3.3 PID controller implementation software:

After estimate continuous and discrete transfer function of DC motors by using DDC system identification approach and design PID controller using PID tune that explained in chapter four, to implement the controller on arduino and see how the DC motors fares with controller .to deploy the controller on hardware, we will use the simulink capability to generate an executable and run it on selected hardware. To test the controller on hardware, we cratered simulink model using block from the arduino support package showed in figure 3.15. We receive desired motor position from serial port and compare it to the measured position from analog input .the position error goes through the PID block which generates a voltage to be sent to the motors. The full experience steps explained in chapter four.

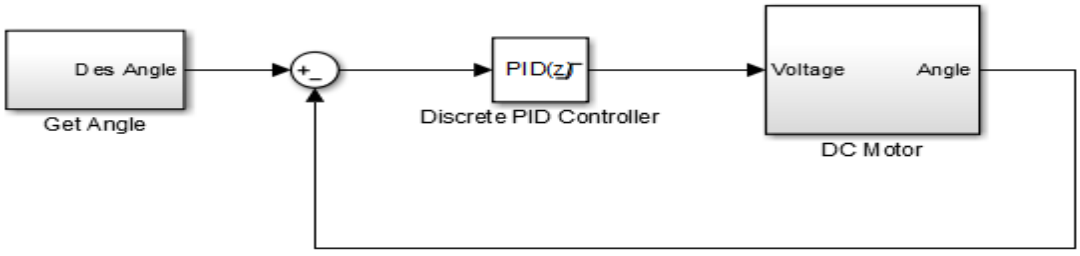


Figure (3.15) SIMULINK model loaded to arduino board

The second simulink code existed in host computer used to transmit input desired angle (reference motor position) to arduino board, and receive the measured angle of DC motor from arduino board throw serial communication

board. The full steps of system PID controller implementation experience and results explained in chapter four. Figure 3.16 shows the simulink model (code) of host computer.

Figure 3.16: Simulink model (code) of host computer

Chapter Four

Simulation experiments and results

4.1Introduction

In this chapter is presented the results of simulating and experimenting with the designed system. The simulation process is a very important step before implementing any project, because it helps the designer to discover the errors and faults in the system before realization. This step is not costly like real implementation and it does not consume a lot of time. Some experiments are carried out in the field after installing this system, so as to check its behavior as a final step. Calculated results from simulation and experiments are discussed here. In order to find the system model of DC motors of robot (mentor), we use the Arduino Uno to collect the data, and we use MATLAB programs to analyze the data and get a model, we apply the PID controller in order to enhance the system performance. Finally we implement the PID controller on arduino to get good control of the position of the DC motors and compare the performance of control algorism with and without PID term, all this work well be discussed in this chapter.

Firstly system identification process: In order to collect the input and output data, we have a potentiometer coupled with the DC motor rotor, that give us variable voltage corresponding to the position of DC motor, we supplied the dc motor by unit step signal of 1V amplitude The output potentiometer voltage was recorded by the arduino card in a PC using MATLA.MATLAB system identification toolbox software has been used to identify a good model by study and analyze the input output data of the motor that captured into computer (arduino card), after that the set of model is tested to choose the perfect model. The following flowchart explains the identification process. Figure 4.1shows the

system identification process.

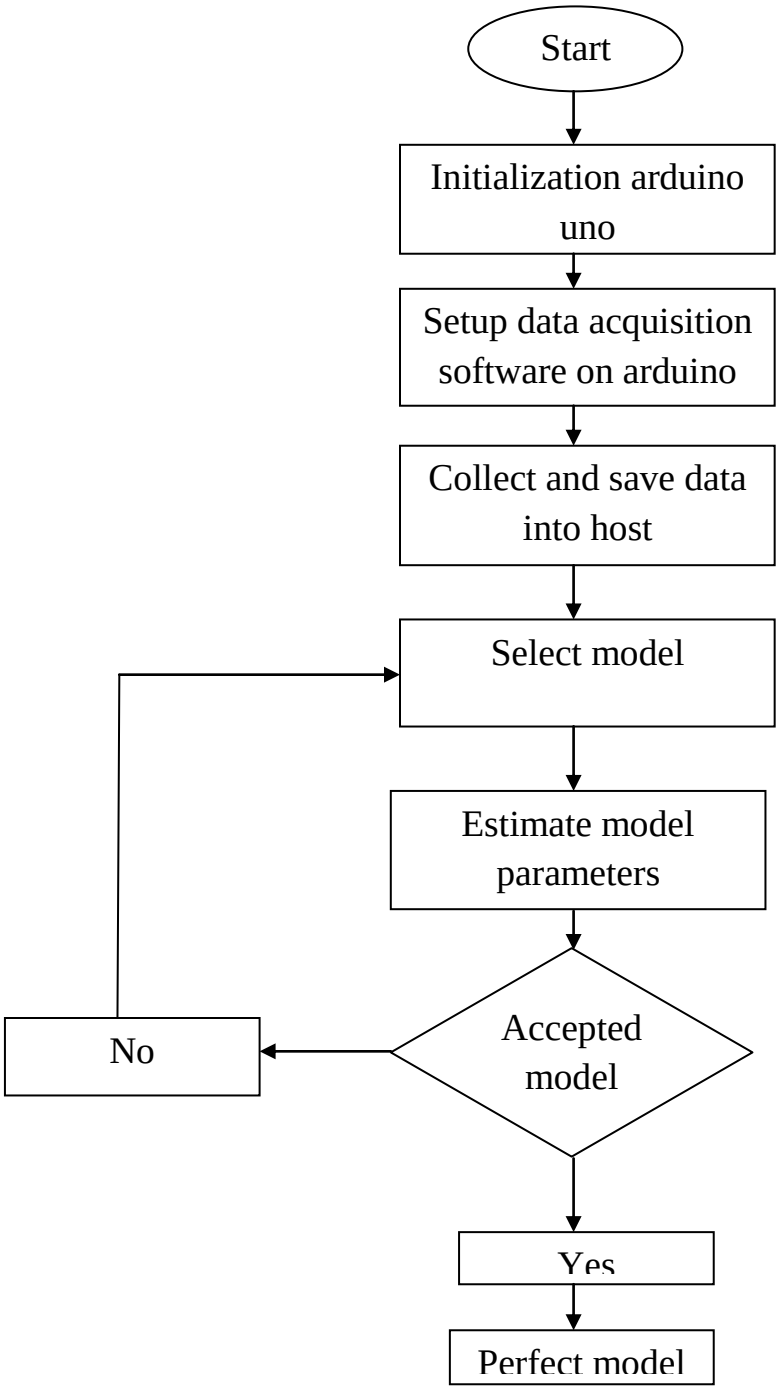


Figure 4.1: System identification process.

4.2System identification

In this thesis there are two system identification experiments for elbow and shoulder DC motors.

4.2.1 Experiment 1(elbow DC motor)

Special code uploaded in arduino card to send the input voltage and receive output voltage, after that by use SIMULINK program the arduino card send the input voltage and output voltage to the computer, SIMULINK program was used to record and saved the input voltage and the corresponding position voltage, Figures (3.9) and (3.10) in chapter three show the block diagram of the hardware setup (how to get data into computer). Figure 4.2 shows system hardware. Figure 4.3 show how to run on target hardware.

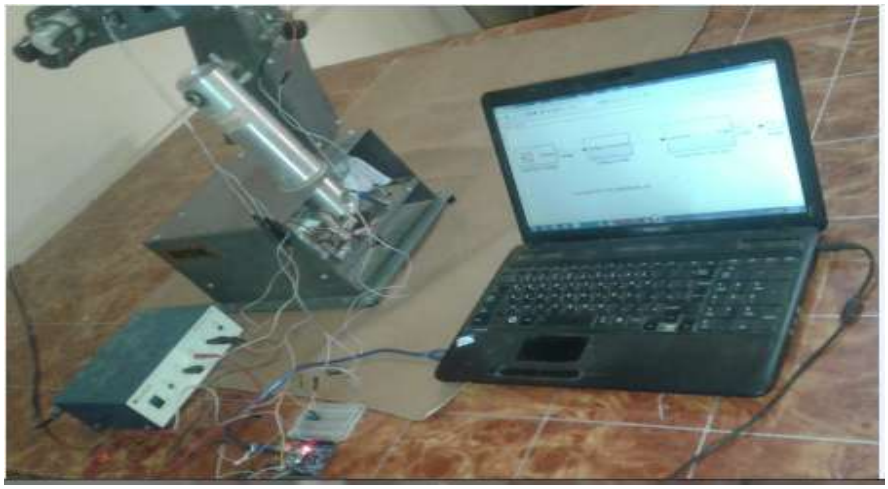


Figure 4.2: system hardware

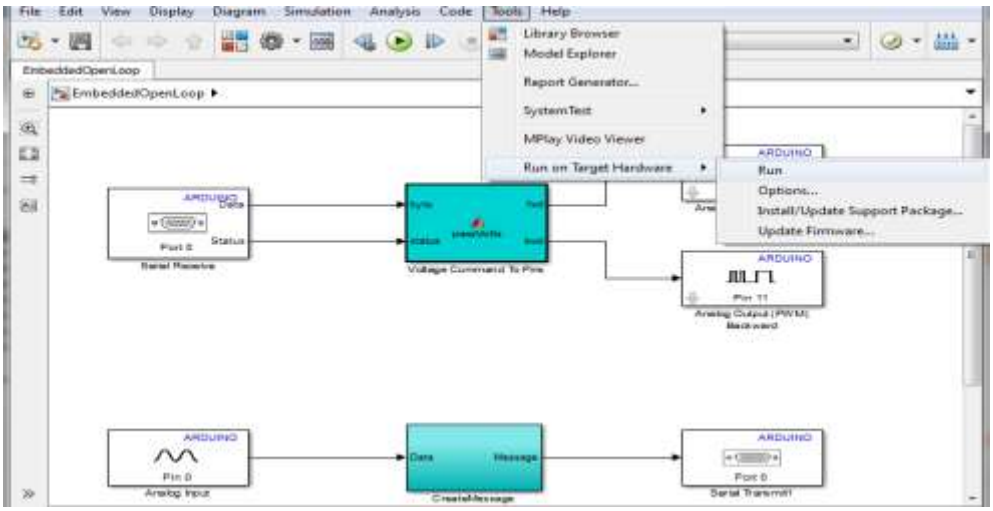


Figure 4.3|:data acquisition simulink model

The main steps for experiment 1as follow:

Step1:

From simulink model of host computer click on reference voltage block as show in Figure 4.4 to select the reference signal as shown in figure 4.5

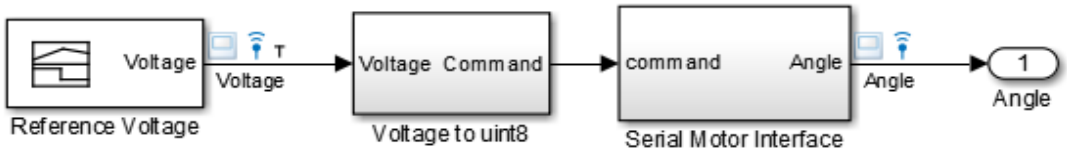


Figure 4.4: Simulink model of host computer

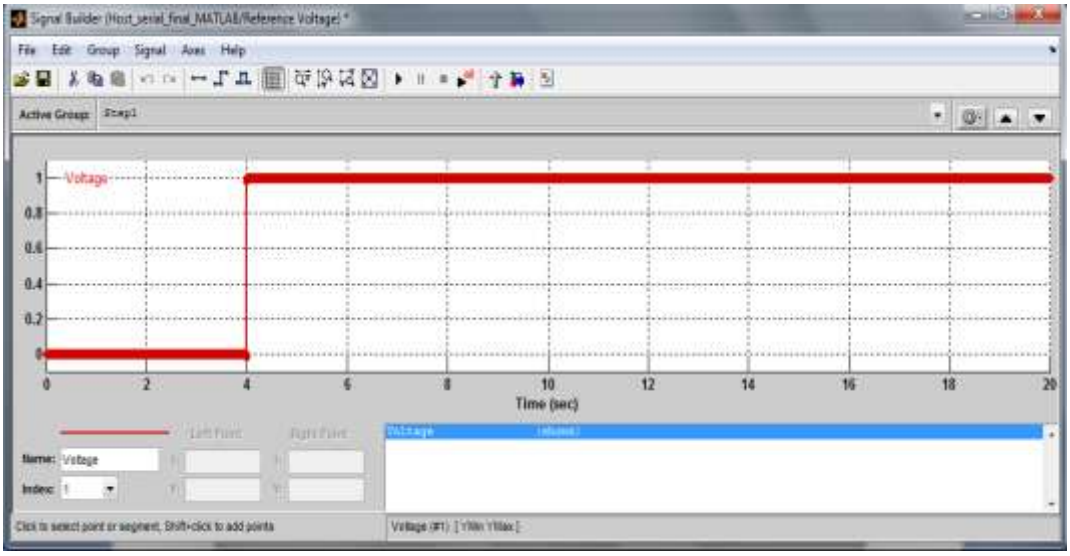


Figure 4.5: reference voltage signal

Step2

Run the simulink model show in Figure 4.6 and look to the result by click angle look point to obtain the output compared with input reference.

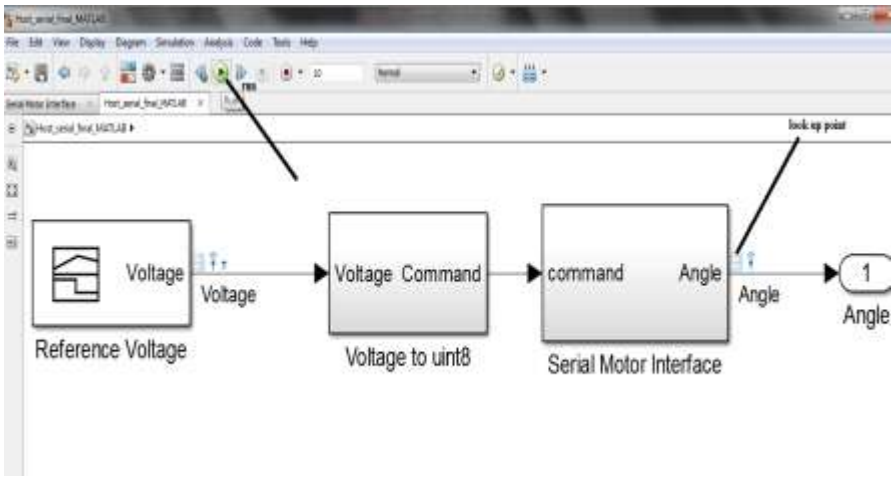


Figure 4.6: run simulink model

Figure 4.7 shows the input voltage and output tracking angle.

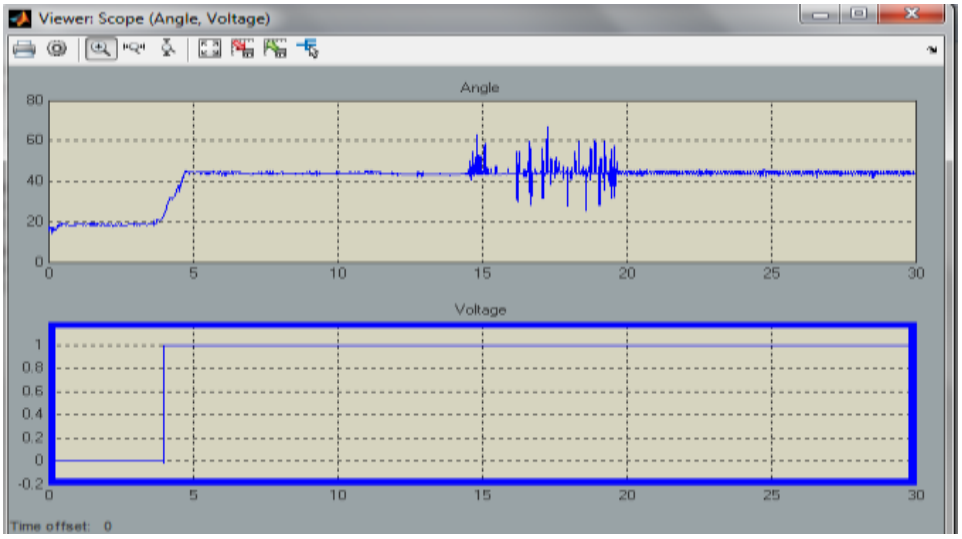


Figure 4.7: input voltage and output tracking angle

Step3

After simulation was done by write command (logsout) in command window>>inter logsout elements appeared as shown in Figure 4.8.

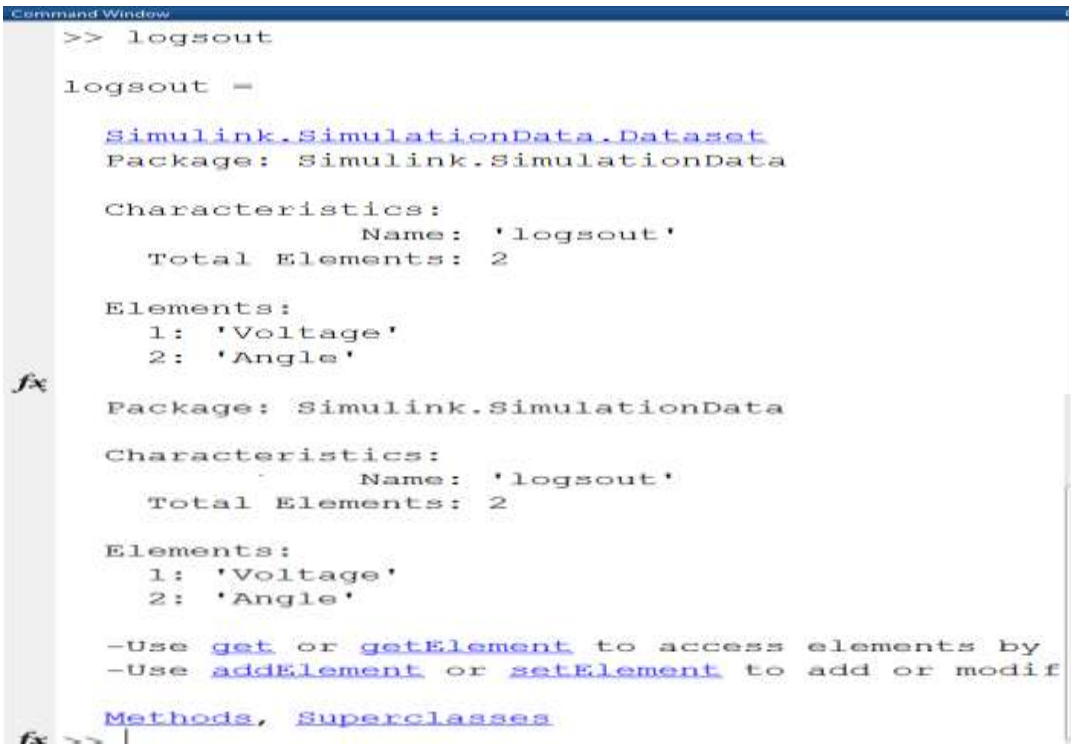


Figure4.8: Logsout command window

Then identified the element 1(voltage), element2 (angle) in command window.

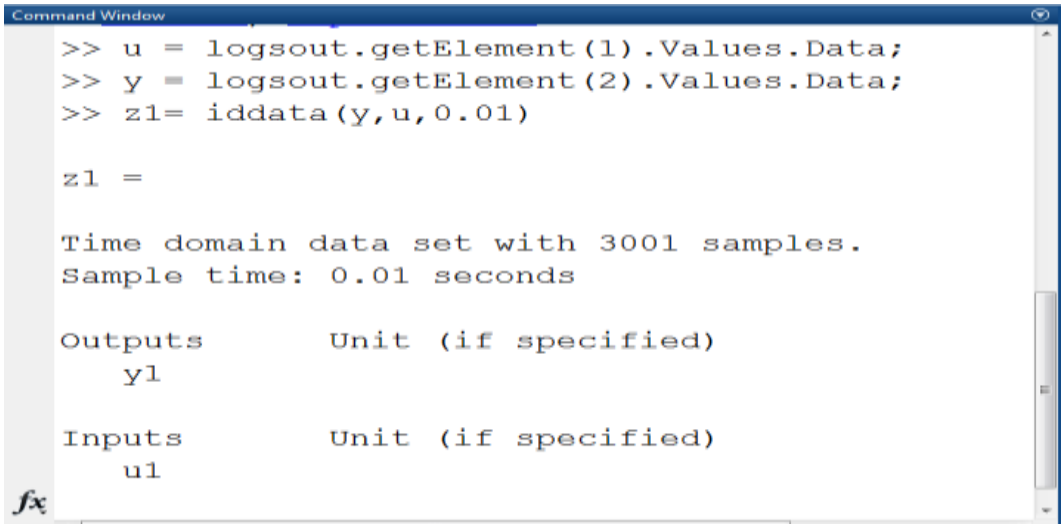


Figure 4.9: Logsout with input\output command window

Step4

by write command(ident) in command window and check inter System Identification Tool window opened, check on import data select data object then

data import window with (IDDATA or IDFRD/FRD) option will open Figure 4.10, inter the object z1and push icon import to obtain the object in system identification tool windows show in Figure4.11 .

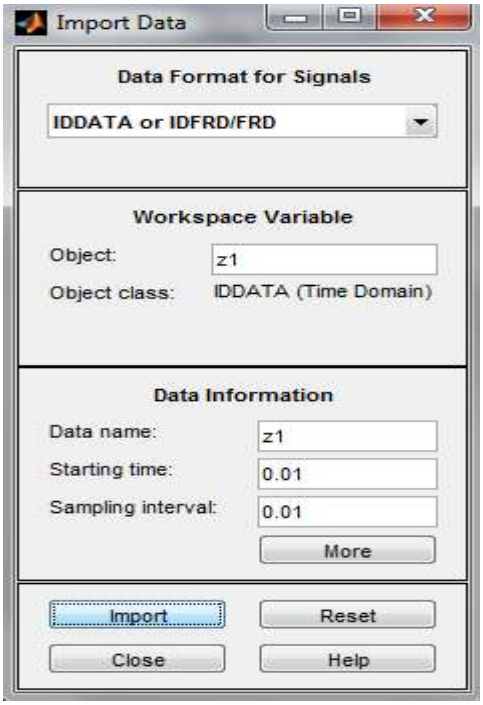


Figure 4.10: import data window

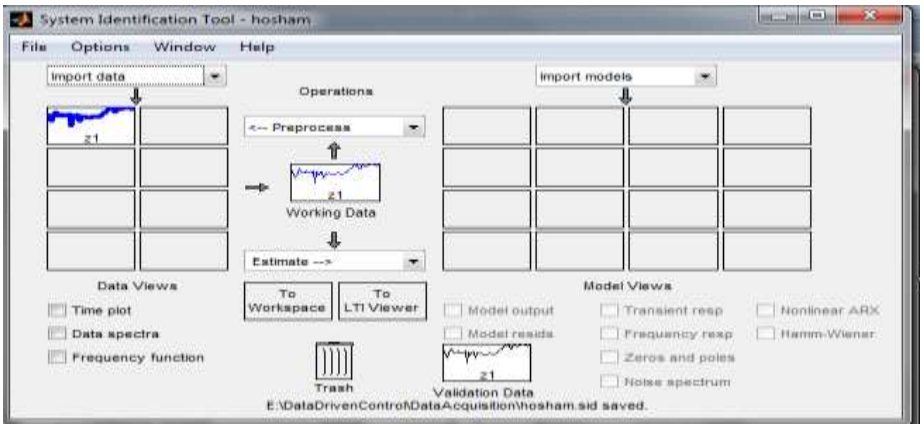


Figure 4.11: system Identification Tool window

In the Import Data dialog box with (time-domain signals) option, we specified the following options:-

- Input — Enter u1 as the name of the input variable
- Output — Enter y1 as the name of the output variable.
- Data name — Change the default name to data. This name labels the data

In the System Identification Tool after the import operation is completed.

- Starting time — Enter 0 as the starting time. This value designates the starting value of the time axis on time plots.
- Sampling interval — Enter 0.01 as the time between successive samples in seconds. This value is the actual sampling interval in the experiment

The Import Data dialog box now resembles the following Figure 4.12.

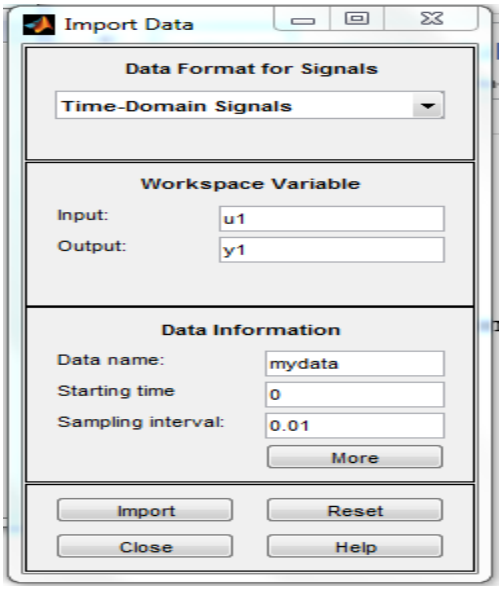


Figure 4.12: Import Data dialog box.

Step5

We used the imported data into two parts as Figure 4.13 and specified the first for model estimation, and the second for model validation. We identified our model from the estimated data, and we used the validation data to validate the model if we have a prefaced model that mean the fitness of model close to

100%, MATLAB program has this characteristic it compare many system together and give us a good one.

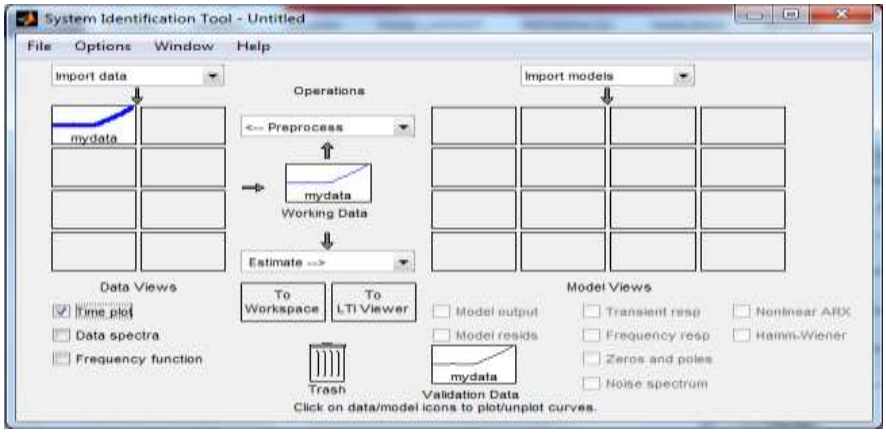


Figure 4.13: imported data into two part (validation, estimation)

By click time plot in figure 4.12 will obtain input voltage and corresponding output position:

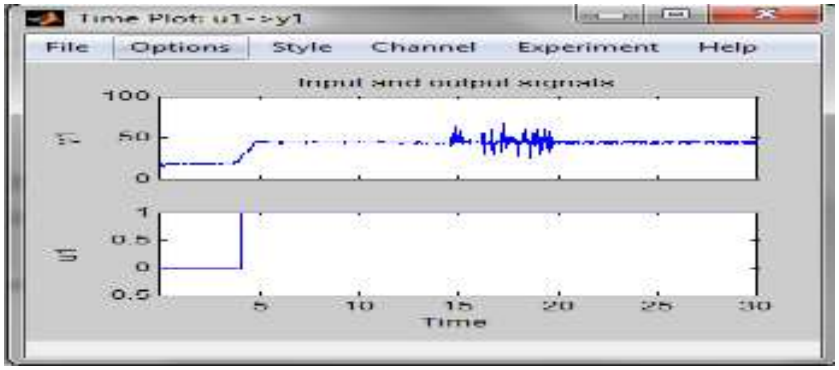


Figure 4.14: time plot input/output signals

Step6:

In the System Identification Tool, select **Estimate** > **transfer function model**, Figure 4.15 shows the estimate dialog box for the way how to estimate the model.

Figure 4.15: the estimate dialog box

By click on transfer function models estimation process will started as flow

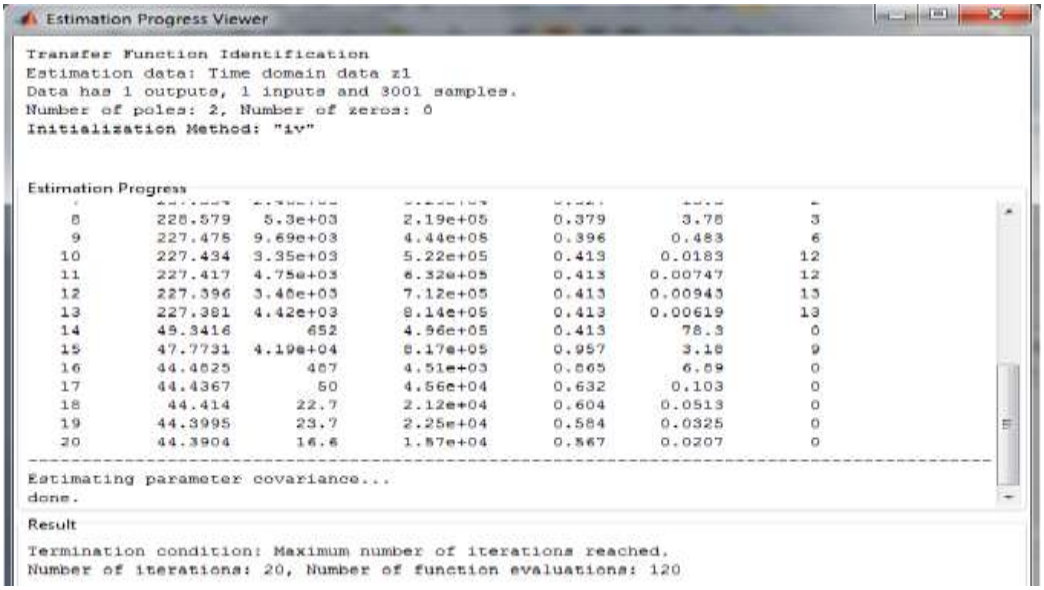


Figure 4.16: estimation progress viewer

By the end the process of estimation progress viewer, system identification tool window figure 4.17 uploaded by tf1(s-domain) and tf2 (discrete time) selected that from transfer function toolbox Figure 4.18.

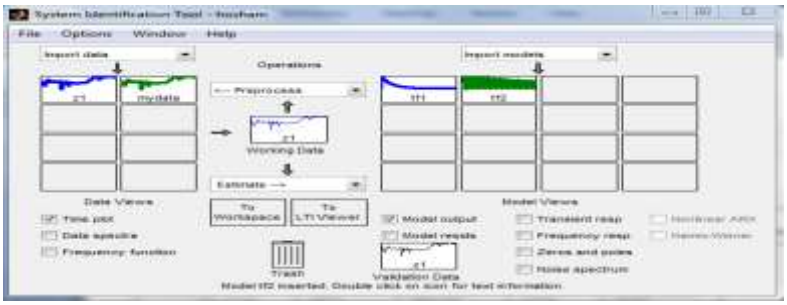


Figure 4.17: System identification tool with transfer function

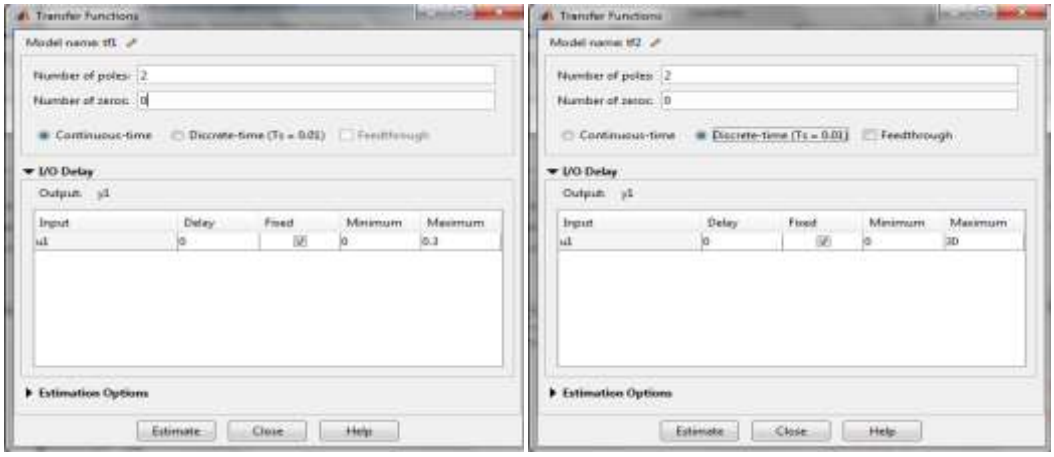


Figure 4.18: transfer function toolbox

By click estimate of tow transfer function toolbox will obtain the transfer function in continues and discrete time domain show in Figure 4.19.

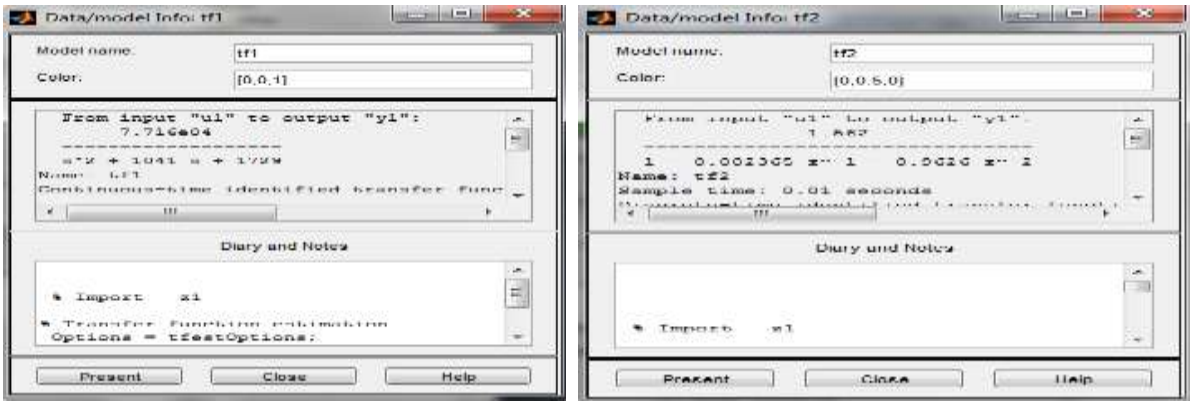


Figure 4.19: estimated transfer function of DC motor in (S and Z) domain

Transfer function of DC motor at z-domain described by equation (4.1)

$$\mathbf{TF} = \frac{\mathbf{1.562}}{\mathbf{1 - 0.002365z^{-1} - 0.9625\ z\hat{}-2}} \qquad \text{equation (4.1)}$$

And Transfer function of DC motor at s-domain described by equation (4.2)

$$\mathbf{TF} = \frac{7.716e04}{\mathbf{s^2 + 10415s + 1729}} \qquad \text{equation (4.2)}$$

4.2.2 Experiment 2(shoulder DC motor)

By using typical steps of system identification experiment 1 we got the flowing results:

Tracking between input voltage and output angle shown in Figure 4.20

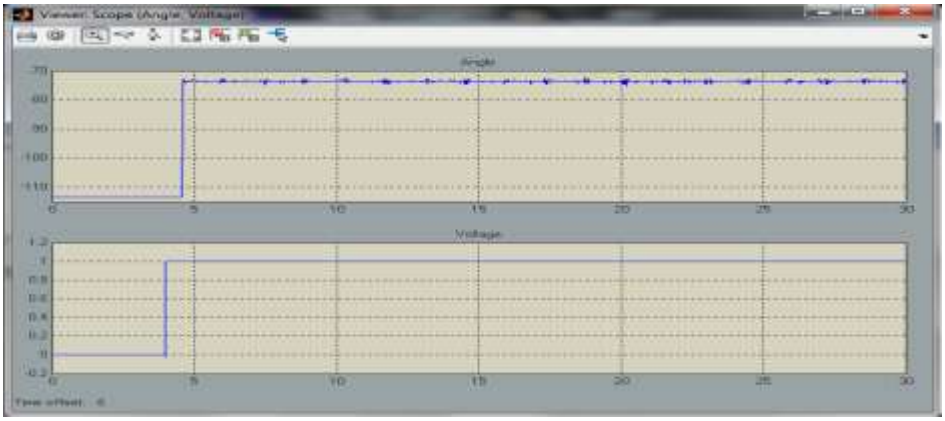


Figure 4.20: input voltage and output tracking angle

Time plot graph shown in Figure 4.22 obtained from system identification tool show in figure 4.21.

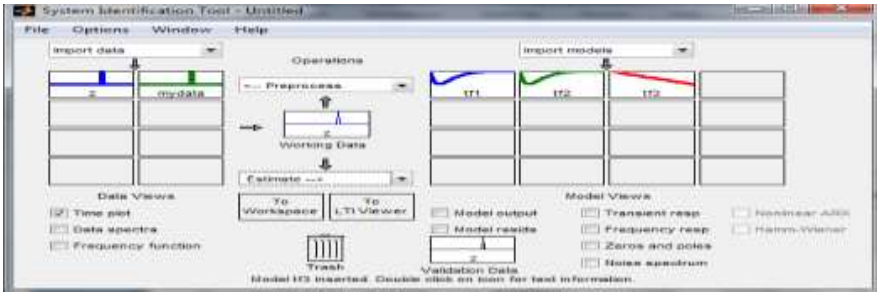


Figure 4.21: system identification tool with transfer function

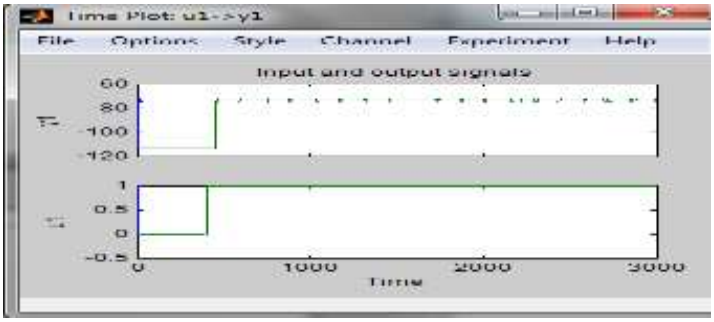


Figure 4.22: Time plot input/output signals

By click estimate of transfer function toolbox will obtain the transfer function in continues and discrete time domain show in Figure 4.23.

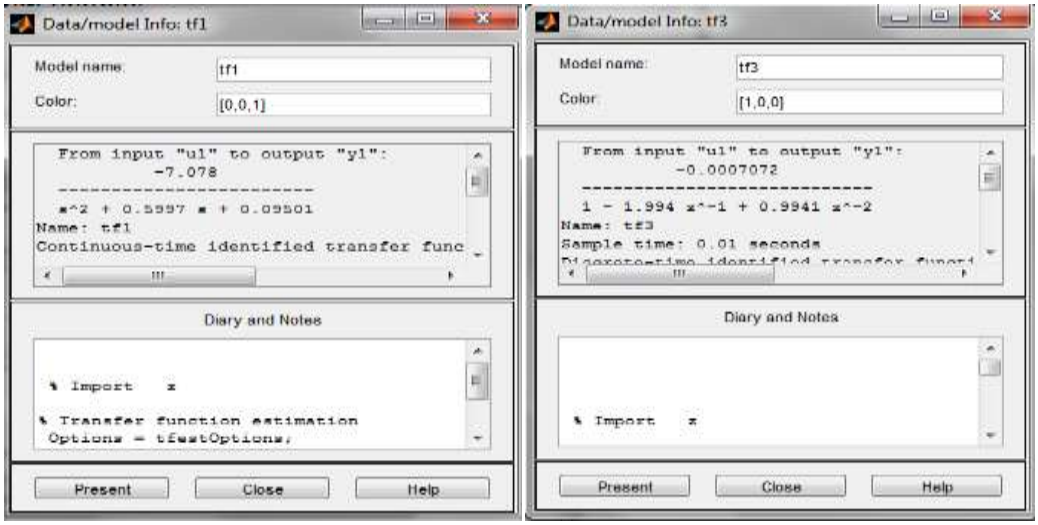


Figure 4.23: Estimated transfer function of DC motor in (S and Z) domain

Transfer function of DC motor at z-domain described by equation (4.3)

$$\mathbf{TF} = \frac{\mathbf{-0.0007072}}{\mathbf{1 - 1.994z^{-1} + 0.9941\ z\hat{}-2}} \qquad \text{equation (4.3)}$$

And Transfer function of DC motor at s-domain described by equation (4.4)

$$\mathbf{TF} = \frac{\mathbf{-7.078}}{\mathbf{s^2 + 0.5997s + 0.09501}} \qquad \text{equation (4.4)}$$

4.3 Design PID controller for elbow DC motor

By check on SIMULINK icon in command window, we opened the SIMULINK window in MATLAB program,

We opened a new project and choose step signal block, PID block, and transfer function block and we connect this component together as follow Figure 4.24.

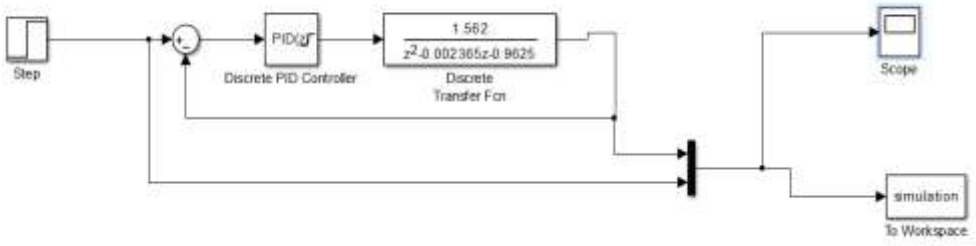


Figure 4.24: Simulink model of PID controller

By click discrete PID controller block we obtain function block parameters of PID controller show in figure 4.25, and by click discrete transfer function block we obtain function block parameters of transfer function show in figure 4.27.

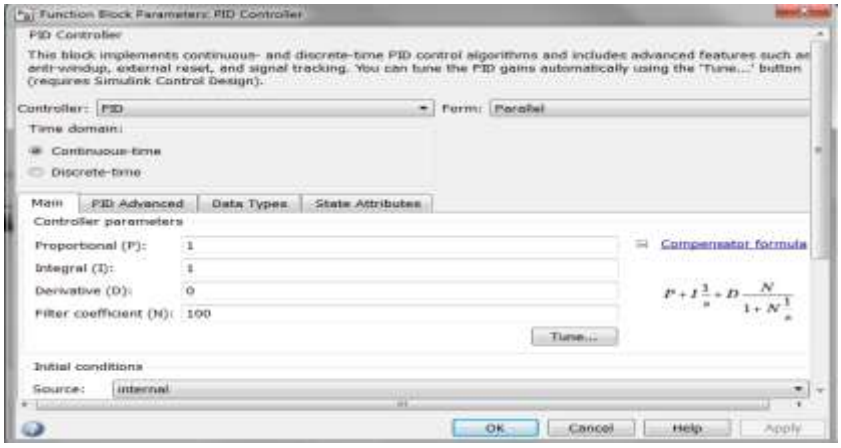


Figure 4.25: function block parameters of PID controller

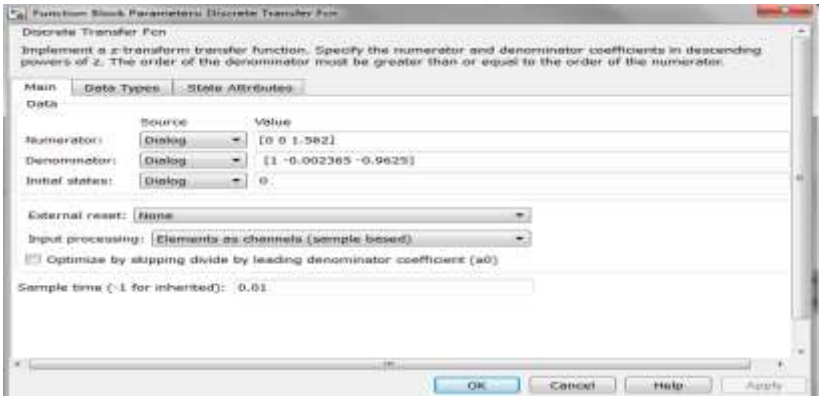


Figure 4.26: function block parameters of transfer function

To obtain optimum PID controller parameters: click icon tune twice in figure 4.25to linearization plant (transfer function) and obtain figure 4.27.

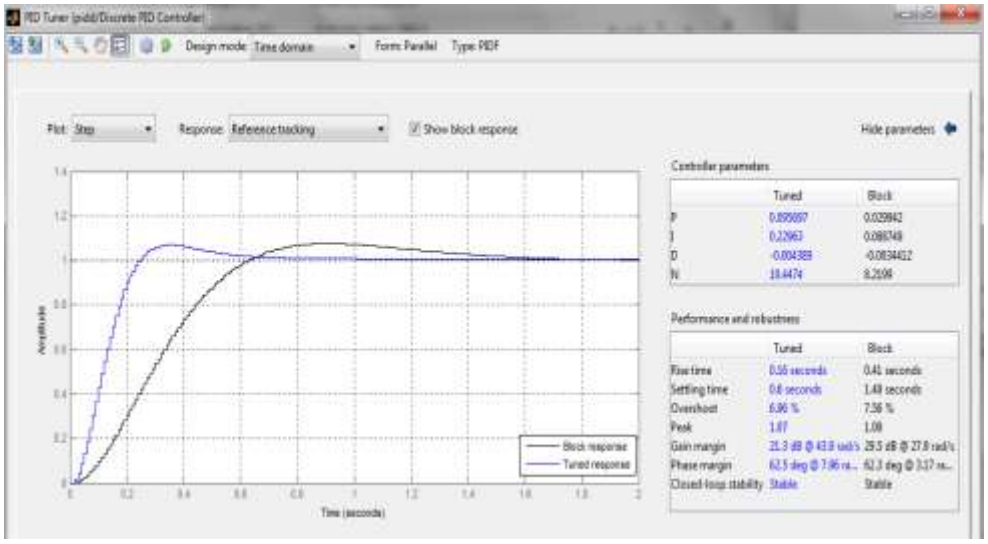


Figure 4.27: tuning PID controller parameters

By moving the Cursor of time response, and monitoring the (Controller parameter, performance and robustness, time response) we choose suitable PID values to control the system as flow:

Table (4.1) result of PID controller suitable values

Controller	**Values**
Proportional gain(K_P)	0.095697
Integral gain(K_I)	0.22693
Derivative gain(K_D)	-0.004389

Simulation

The goal of this thesis is show how each of K_p, K_i and K_d contributes to obtain Fast rise time, Minimum overshoot, and No steady-state error for system

of DC motor that drive the robot arm. Figure 4.28 shows elbow DC motor response without PID controller.

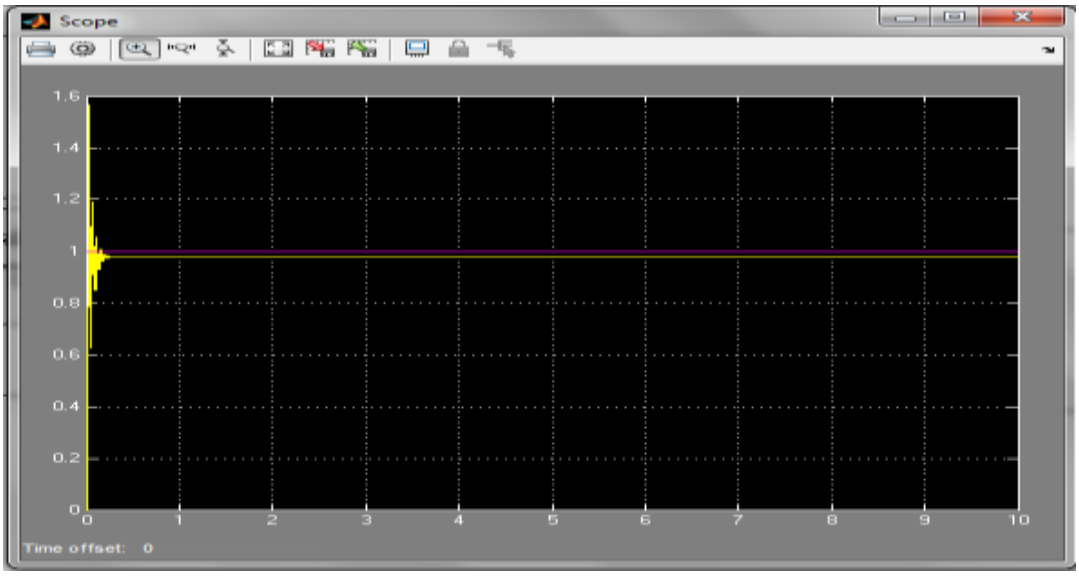


Figure 4.28: DC motor (elbow) response without PID controller

And Figure 4.29shows elbow DC motor reponse with PID controller.

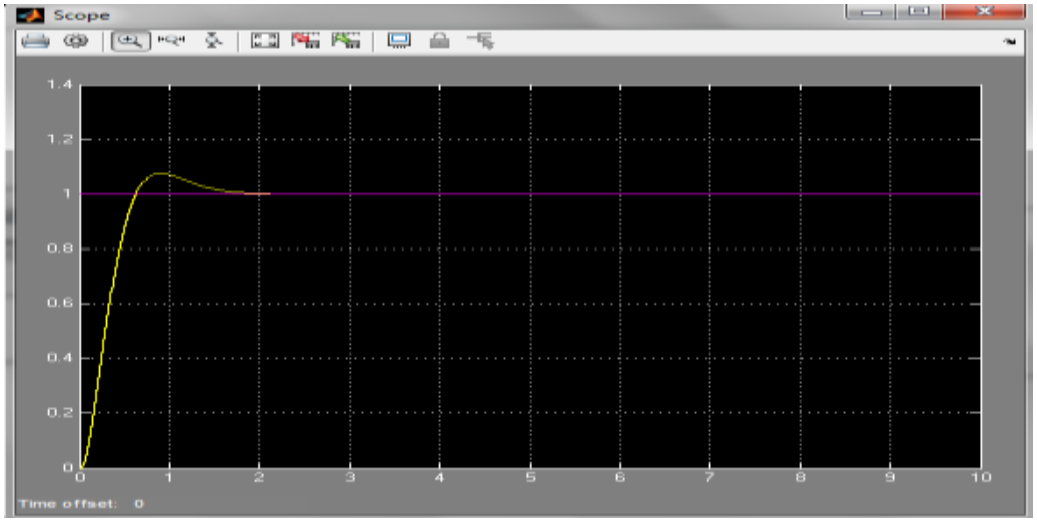


Figure 4.29: DC motor(shoulder) response with PID controller

4.4Design PID controller for shoulder DC motor

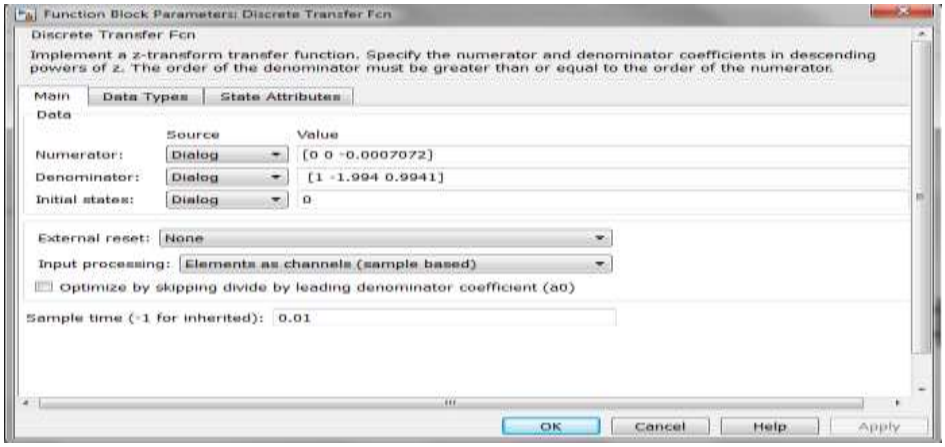


Figure (4.30) function block parameters of transfer function

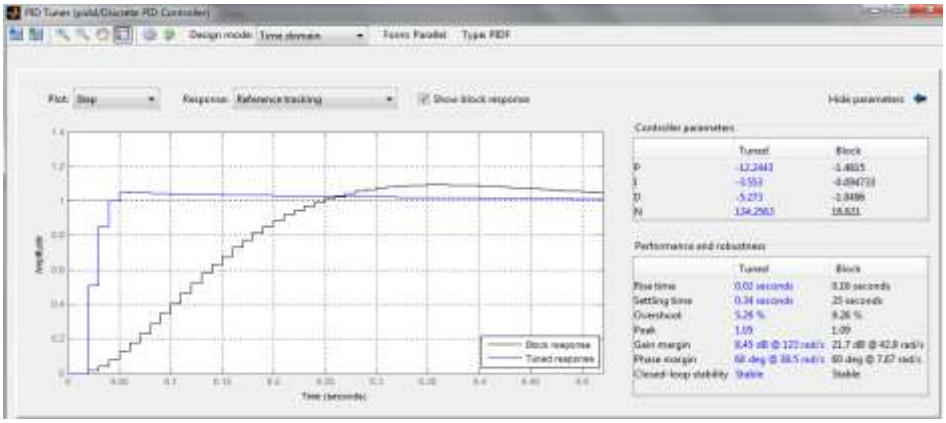


Figure (4.31) tuning PID controller parameters

By moving the Cursor of time response, and monitoring the (Controller parameter, performance and robustness, time response) we choose suitable PID values to control the system as flow:

Table (4.2) result of PID controller suitable values

Controller	**Values**
Proportional gain(K_P)	-12.2443
Integral gain(K_I)	-3.553
Derivative gain(K_D)	-5.273

Simulation

Figure 4.32 shows simulink model of PID controller.

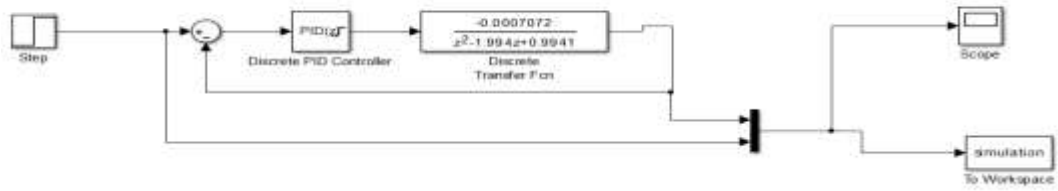


Figure 4.32:Simulink model of PID controller

Figure 4.33shows shoulder DC motor response without PID controller.

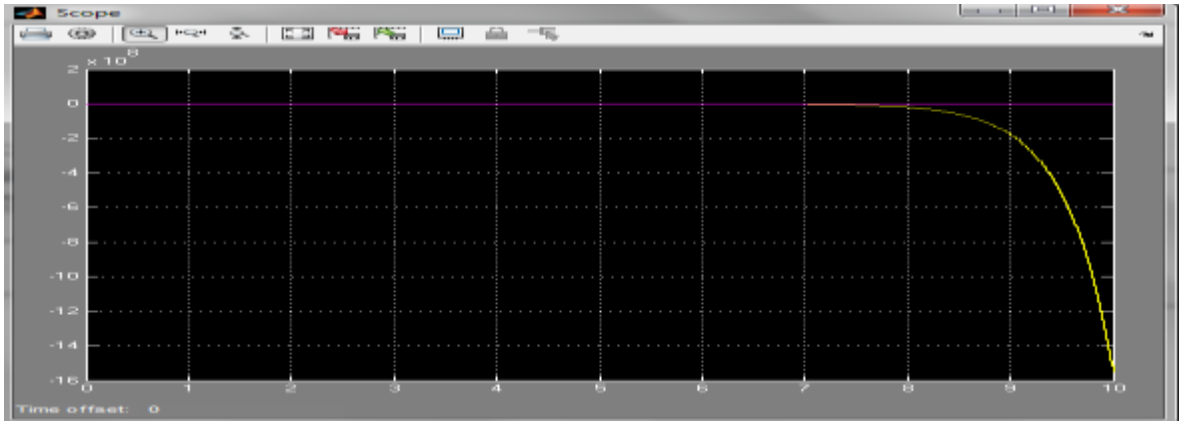


Figure 4.33: DC motor response before PID controller

Figure 4.34shows the shouder DC motor response with PID controller.

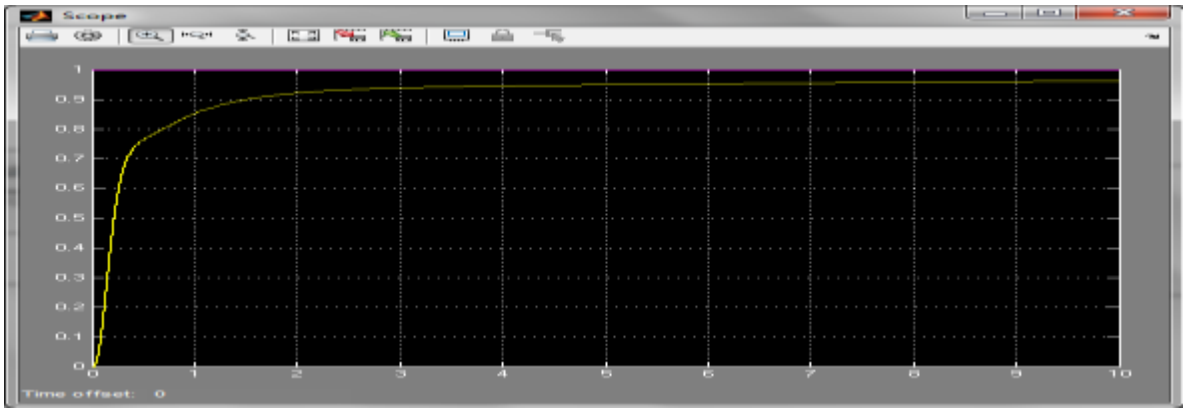


Figure 4.34: Shoulder DC motor response with PID controller

4.5Real time testing for elbow DC motor

After tuning and obtained optimum parameters of PID controller and compare between the system before and after PID controller in simulation, the last step is to implement PID controller on Hardware system .To control the position of DC motor of robot arm we need to control the input voltage that supply the motor , the control process of input voltage has been done by compare the input voltage with output voltage that come from output potentiometer , we put the desired position and the PID controller transmit control signal as PWM to rotate the motor to corresponding position , arduino card read the value of current position of motor as voltage from the output potentiometer and send this value to controller, controller compare the output voltage with voltage of desired position and give suitable output signal corresponding to difference between them ,if difference equal to zero signal equal to zero, The PID algorism output the control signal that rotate the motor to desired position.

Also special code (simulink) uploaded in arduino card to send the input angle and receive output angle to DC motor, and special code on host computer use to send the reference position data (angle) and collect the actual position feedback and check how motor track the reference, Figures 3.34 and Figure (3.35) in chapter three shows tow simulink models .

Step1

Run simulink model in figure 3.34 on arduino Uno board.

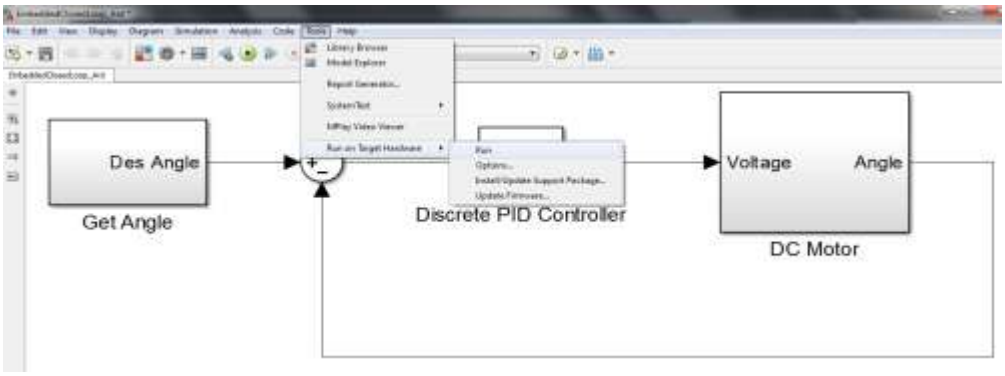


Figure 3.35: Simulink on arduino Uno board

Step2

Runs the simulink model showed in figure3.35, click on gain slider to produce deferent angles between (15to -15) degree and look to the result by click angle look point to obtain the output compared with input reference.

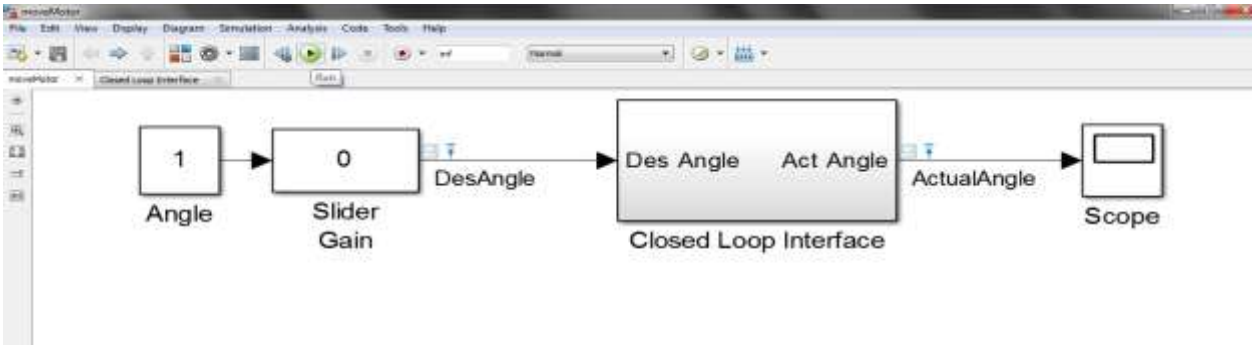


Figure 4.36: Host computer simulink

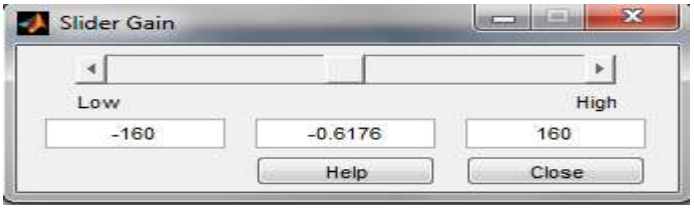


Figure 4.37: Slider gain controller

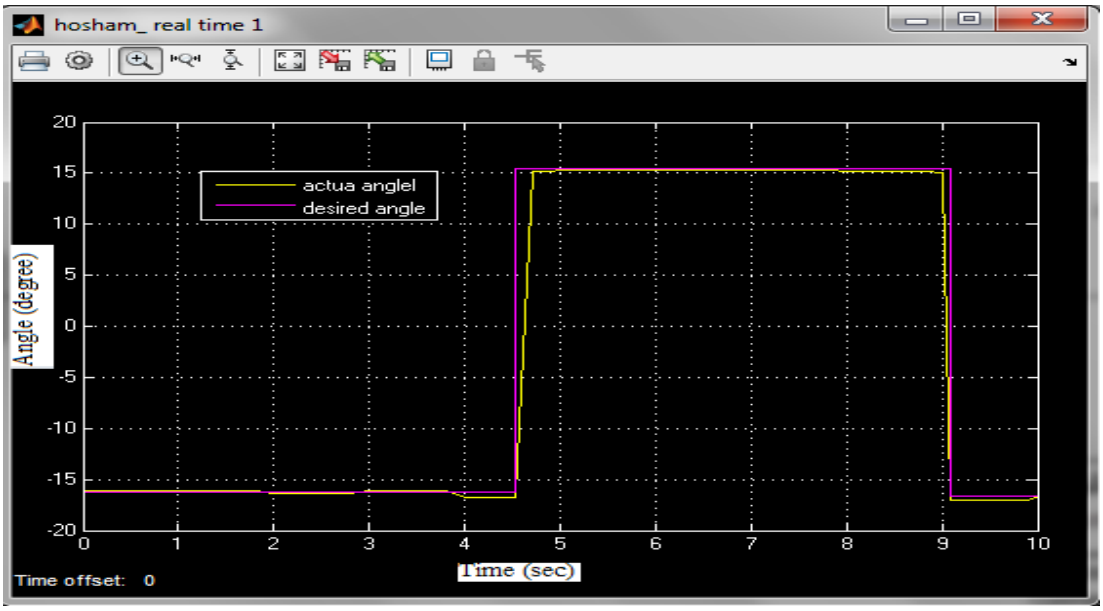


Figure 4.38: Desired angle and actual angle (hardware) graph

4.6 Real time testing for shoulder DC motor

By using typical steps of real time testing(1) and click on gain slider to produce deferent angles between (40to -110) degree and look to the result by click angle look point to obtain the output compared with input reference.

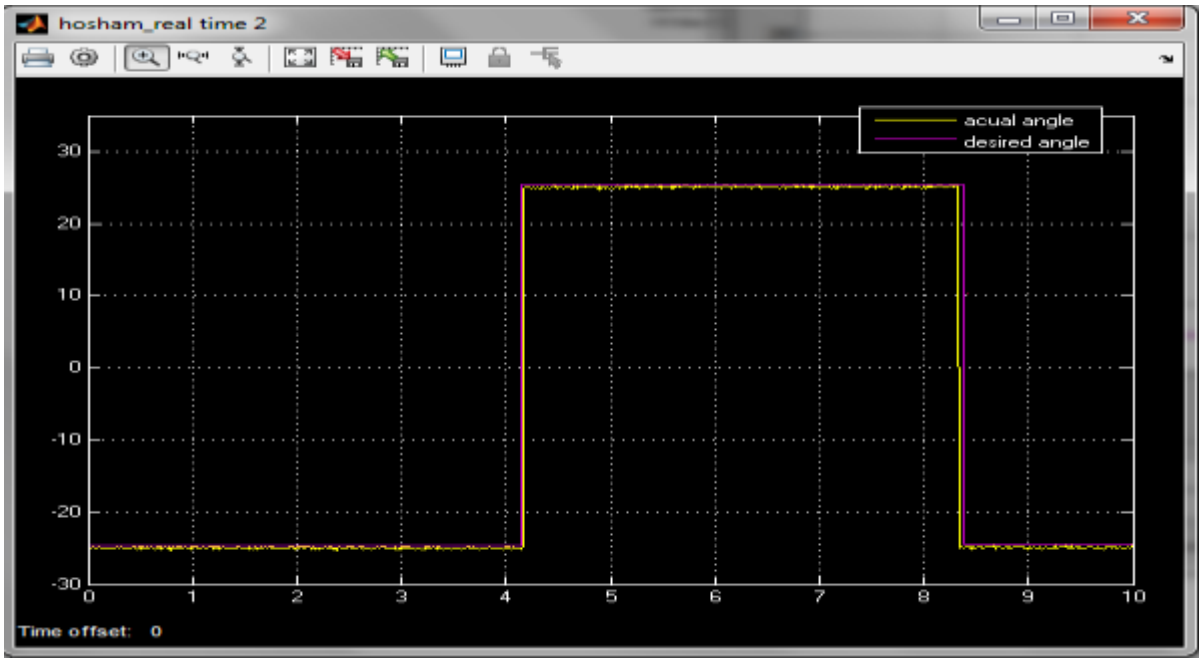


Figure 4.39: desired and actual angles graph of reference angle

Chapter five

Conclusions and Recommendations

5-1 Conclusion

Our robot developed with an implementation of Arduino Uno board, can be guided with two degrees of freedom using PID controller. We used geared motors for their high torque at low speeds. These design tradeoffs were chosen for the envisioned target application of robots interacting with unstructured environments such as a typical home or workplace, where the safety of intrinsic mechanical compliance is an important design consideration. These robots have a wide range of industrial and medical applications (pick and place robot).

The procedure of building robotic arm using PID controller in Arduino Uno with host computer programs and DC motors with the help of potentiometers has been successfully satisfied. The building procedure consists of hardware and software design, implementation, and programming.

5-2 Recommendations

As the future work of the developed robotic arm:

- Complete the design and implement PID controller for other axis0, axis3 and axis4.
- Design and implementing fuzzy PID controller and compare it with classical PID controller.
- A voice-controlled robot can be considered where the user can simply control the robot by giving voice commands such as giving directions (up, down, left, right) and actions (grasp, drop, rotate left, and rotate right).
- Wireless communication between the user and the robot for remote control: thus provide safety for the user and allowing him to move in an even easier unrestricted manner.
- Add one axis control to developed pick and place robot to handling robot.

References

[1] M. H. Rashid, “Power Electronics Circuits, Devices and Applications”, pp. 640–781,2004.

[2] W. C. M. yongbin, L. Yongxin, “Design of Parameters Self-tuning Fuzzy PID Control for DC Motor,” pp. 345–348, 2010.

[3] R. G. Kanojiya and P. M. Meshram, “Optimal Tuning of PI Controller for Speed Control of DC Motor Drive using Particle Swarm Optimization”, 2012.

[4] V. M. V. Rao, “Performance Analysis of Speed Control Of Dc Motor Using P, PI, PD and PID Controllers,” Vol. 2, No. 5, pp. 60–66, 2013.

[5] M. S. Najib, M. S. Jadin, R. M. Taufika, and R. Ismail, “Design and Implementation of PID Controller in Programmable Logic Controller for DC Motor Position Control of the Conveyor System”, pp. 266–270, 2007.

[6] J. M. Neto, S. Paladin, C. E. Pereira, and R. Marcelino, “Remote Educational Experiment Applied To Electrical Engineering”, 2012.

[7] Z .Othman, “High Efficiency” Dual Axis Solar Tracking Development Using Arduino”,pp.43-47,2013.

[8] J. Antonio, “Automatic Control for Laboratory Sterilization Process based on Arduino Hardware”,pp. 130–133, 2012.

[9] A.Z. Alassar, I.M. Abuhadrous and H.A. Elaydi “Control of 5DOF Robot ArmUsing PID Controller with Feedforward Compensation”, Accepted in the 2ndConference on Computer and Automation Engineering, Singapore, 5. Dec. 2009.

[10] S.G. Anavatti, S.A. Salman and J.Y. Choi, “Fuzzy + PID Controller for Robot Manipulator”, International Conference on Computational Intelligence for Modelling Control and Automation, pp. 75, Dec. 2006.

[11] Shiuh-Jer, H. and L. Ji-Shin ."A stable Self-Organizing Fuzzy Controller for Robotic Motion Control", Industrial Electronics, IEEE Transactions on 47(2),PP421-428, 2000.

[12] Kiam Heong, A.,"PID Control System Analysis, Design, and Technology", Control Systems Technology, IEEE Transactions on 13(4),PP559-576, 2005.

[13]Visioli, “Tuning of PID Controllers with Fuzzy Logic”, IEEE Proceedings Control Theory and Applications, Vol. 148, No. 1, pp. 1-8, 2001.

[14] Mohmmed Huesien, "Design Controller System for Rbotic Arm", Vo1. 6, No. 4, pp449–463, Nov, 2016.

APPENDIX A
ARDUINO MODEL AND CODE COLLECTING INPUT/OUTPUT DATA

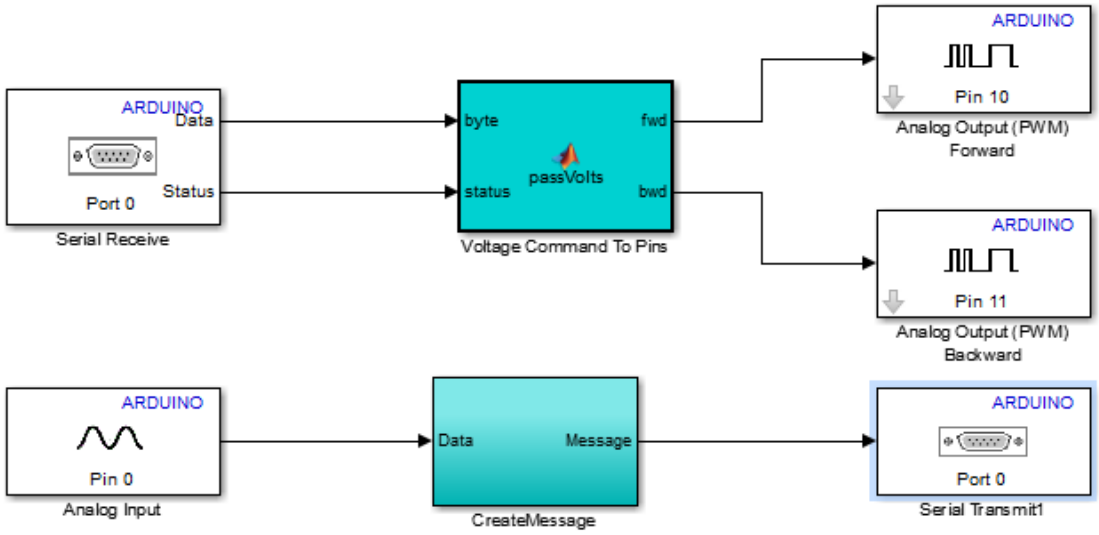


Figure A.1: Arduino code to collect input output data

Voltage command to pins code :

```
Function [fwd,bwd] = pass Volts (byte, status)
%#codegen
persistent last
if isempty(last) last = int16(0);
end
if (byte>=0 && status == 1)
volt = 2*(int16(byte)-127);
last = volt;
lse 71
```

```
volt = last;
end
if volt>0
fwd = uint8(volt) ;
bwd = uint8 (0) ;
else
fwd = uint8(0) ;
bwd = uint8 (-volt) ;
end
```

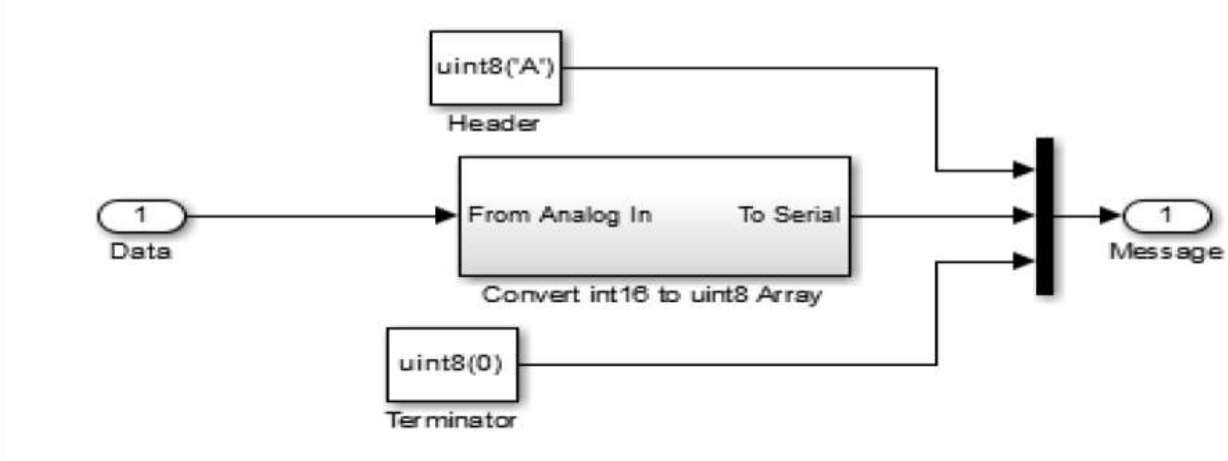


Figure A.2: Create Message

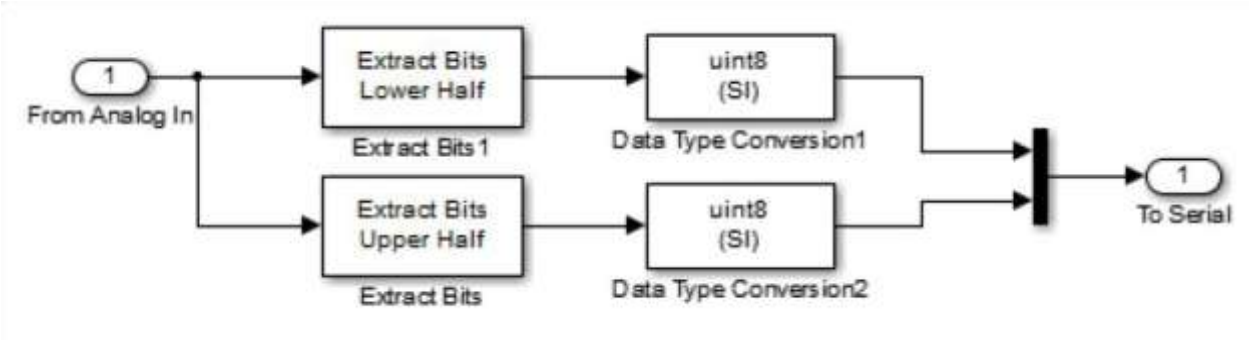


Figure A.3: Convert int16 to uint8 Array

APPENDIX B
SIMULINK MO TO THE MOTOR TO GET INPUT/OUTPUT DATA

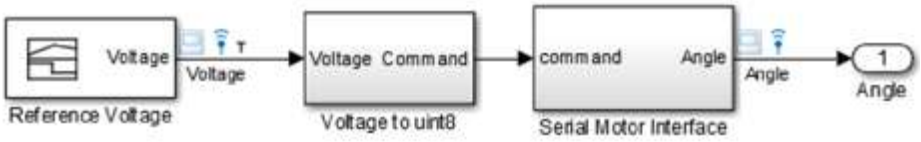


Figure B.1: SIMULINK model to the motor to get input and output data.

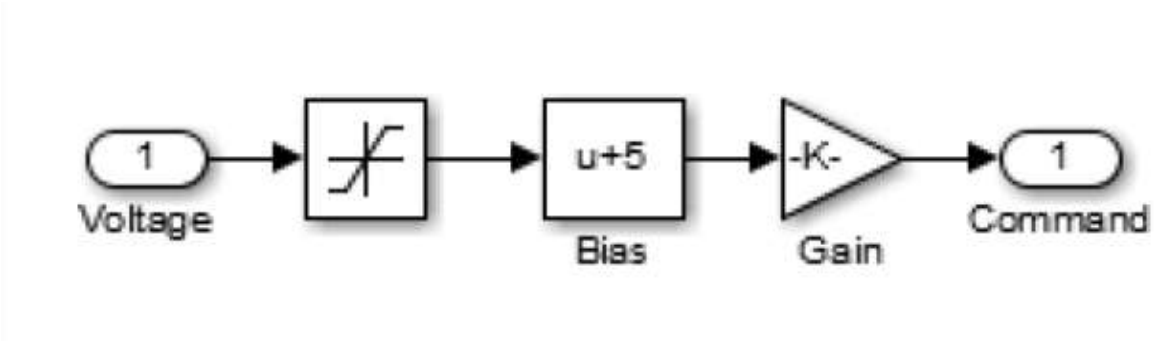


Figure B.2: voltage to uint8

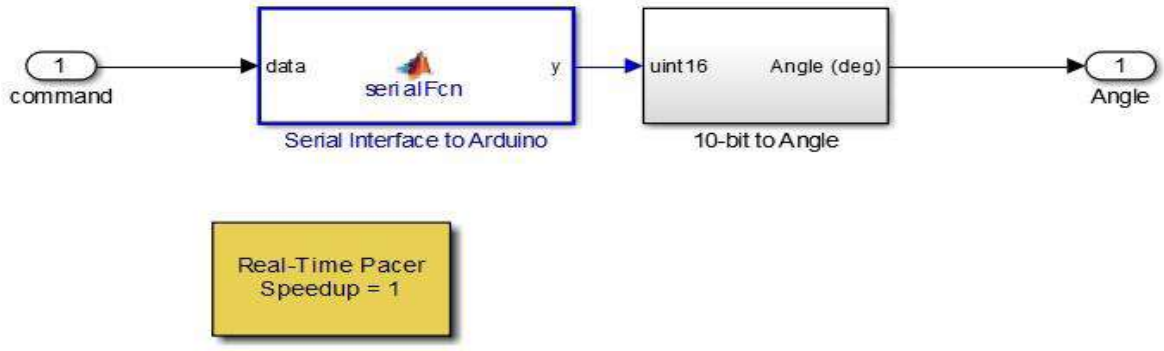


Figure B.3: Serial Motor Interface

Serial receive and transmit code:

```
function y = serialFcn(data)
coder.extrinsic('fread','fwrite','instrfind')
persistent s last 73
% Declaring parameters
```

```
HEADER = 'A';
PACKET_SIZE = 4;
packet = zeros(PACKET_SIZE,1,'int16');
% Initializing the serial object
if isempty(s) s = instrfind;
 last = int16(0); end
% Write data to the serial port fwrite(s,data,'uint8');
% Read data from the serial port
packet = int16(fread(s, PACKET_SIZE, 'uint8'));
% Check for header and convert two bytes to single int16
if (packet(1) ~= HEADER)
y = last; else
y = 2^8*packet(3)+packet(2); last = y;
end
%y=0
```

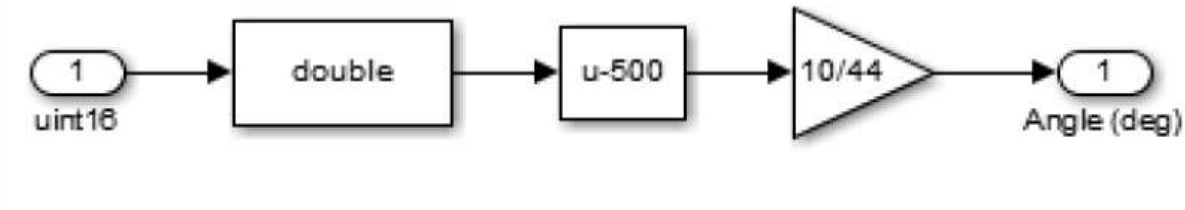


Figure B.4:10-bit to Angle

APPENDIX C
Controller processing

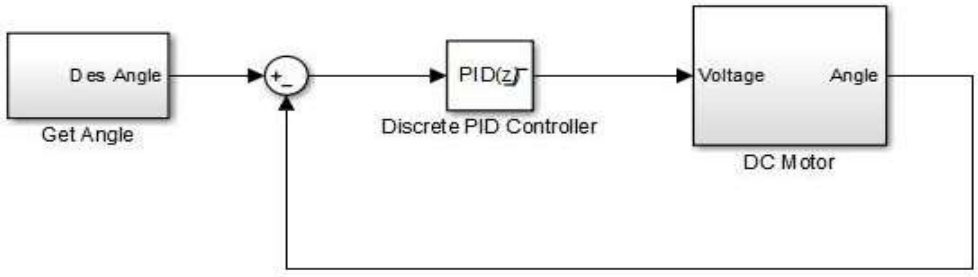


Figure C.1: PID code simulink

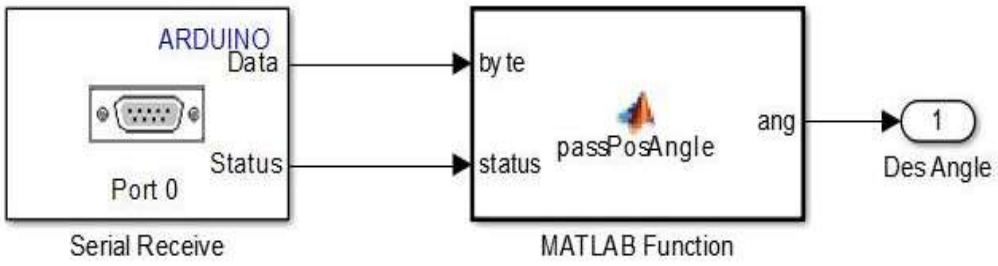


Figure C.2: Get angle simulink

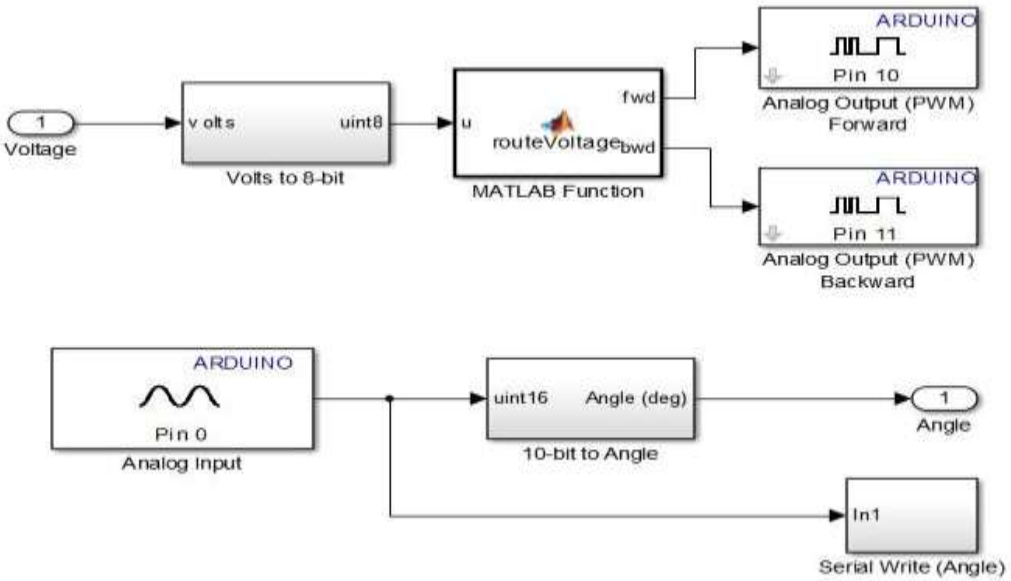


Figure C.3: Block of DC motor

MATLAB Function

```
function [fwd,bwd] = routeVoltage(u)
%#eml if u>0
fwd = uint8(u); bwd = uint8(0);
else
fwd = uint8(0); bwd = uint8(-u);
end
```

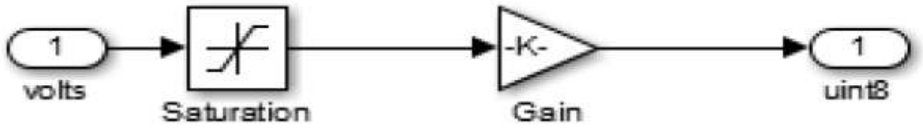


Figure C.4: Volts to 8-bit

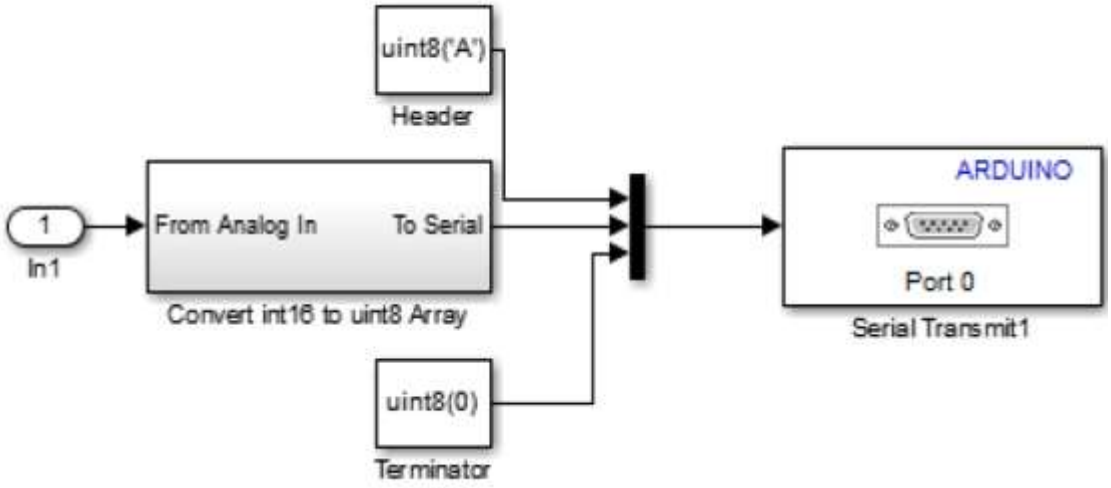


Figure C.5: Serial Write (Angle)

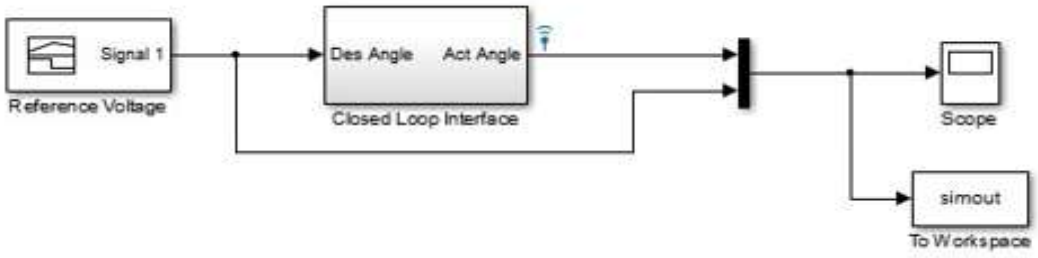


Figure C.6: Simulink code to turn motor

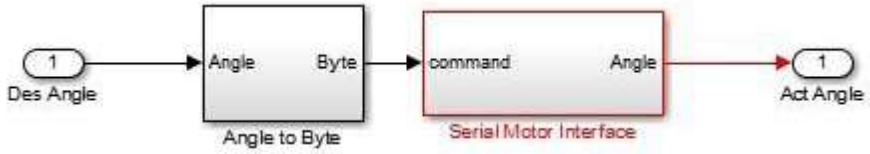


Figure C.7: Angle to Byte

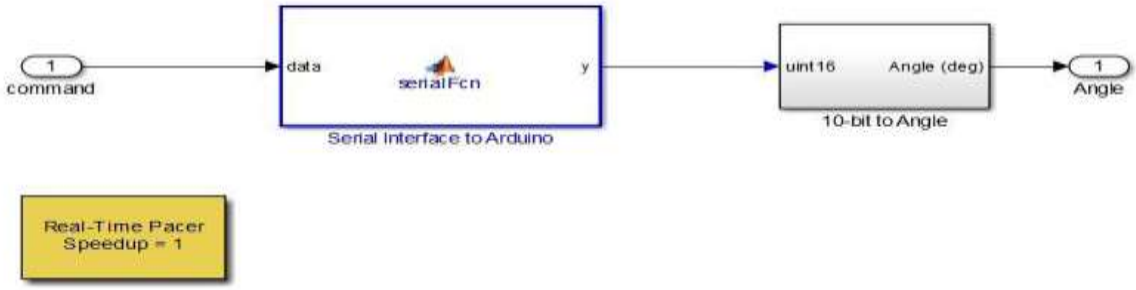


Figure C.8: Serial Motor Interface

Figure C.9: 10 –bit to angle

Serial transmit code

```
function y = serialFcn(data)
coder.extrinsic('fread','fwrite','instrfind')
persistent s last
% Declaring parameters
HEADER = 'A';
 PACKET_SIZE = 4;
packet = zeros(PACKET_SIZE,1,'int16');
% Initializing the serial object if isempty(s)
 s = instrfind;
last = int16(0);
end
% Write data to the serial port
fwrite(s,data,'uint8');
% Read data from the serial port packet =
int16(fread(s, PACKET_SIZE, 'uint8'));
% Check for header and convert two bytes to single
int16
if (packet(1) ~= HEADER) y = last;
 else
y = 2^8*packet(3)+packet(2);
last = y;
end
% y= 0;
```