SHEERAZ ALI MEMON

ATILIM UNIVERSITY 2022

SIMULATION FOR CONTROL SYSTEM OF DRAWWORKS AND ITS IMPLEMENTATION USING PLC FOR OIL AND GAS DRILLING

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

SHEERAZ ALI MEMON

A MASTER OF SCIENCE THESIS
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2022

SIMULATION FOR CONTROL SYSTEM OF DRAWWORKS AND ITS IMPLEMENTATION USING PLC FOR OIL AND GAS DRILLING

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
ATILIM UNIVERSITY

BY

SHEERAZ ALI MEMON

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

SEPTEMBER 2022

Approval of the Graduate School of Natural and Applied Sciences, Atilim University.

Prof. Dr. Ender KESKİNKILIÇ
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of **Master of Science in Electrical and Electronics Engineering Atilim University**.

Prof. Dr. Reşat Özgür DORUK
Head of Department

This is to certify that we have read the thesis "SIMULATION FOR CONTROL SYSTEM OF DRAWWORKS AND ITS IMPLEMENTATION USING PLC FOR OIL AND GAS DRILLING" submitted by "SHEERAZ ALI MEMON" and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Mehmet Efe ÖZBEK
Supervisor

Asst. Prof. Dr. Yaser DALVEREN
Electrical and Electronics Engineering, Atilim University

Asst. Prof. Dr. Mehmet Efe ÖZBEK
Electrical and Electronics Engineering, Atilim University

Assoc. Prof. ENVER ÇAVUŞ
Electrical and Electronics Engineering,
Ankara Yıldırım Beyazıt University

Date: *02/09/2022*

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : SHEERAZ ALI MEMON

Signature :

ABSTRACT

SIMULATION FOR CONTROL SYSTEM OF DRAWWORKS AND ITS IMPLEMENTATION USING PLC FOR OIL AND GAS

Memon, Sheeraz Ali

MSc., Department of Electrical and Electronics Engineering

Supervisor: Asst. Prof. Dr. Mehmet Efe ÖZBEK

August 2022, 73 pages

Drawworks is one of main component of drilling rig. To control drawworks it requires skill and experience. Driller that controls the drawworks has very difficult position for work because it is near to drilling rig. So, chances are more for unsafety. Failure of drawworks system could risk the life of driller if any error occurred. I have simulated drawworks control system and implemented the simulation to prototype model of drawworks control system. In prototype model of drawworks system for safety of driller and drawworks deadman button, ground fault error detector, deadband region has been added. To avoid jerks joystick analog signal has been smooth. Weight is one of most important parameters in drilling. Weight equilibrium have been done for travelling block if there is any change in weight, travelling block would move upward or downward according to weight. So, by applying these safety features drawworks system can be prevent from errors and jerks, this way safety is increased for drawworks system as well as for driller because of its work position near to the drilling rig. MATLAB and Twincat3 software has been used for simulation and implementation.

Keywords: Drawworks, vibrations, weight equilibrium, deadband region, safety.

ÖZ

SİMÜLASYONU DRAWWORKS KONTROL SISTEMI. PETROL VE GAZ İÇİN PLC KULLANARAK ONUN UYGULAMA

Memon, Sheeraz Ali

Master, Elekrik Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Dr. Öğr.Üyesi Mehmet Efe ÖZBEK

Ağustos 2022, 73 sayfa

Drawworks, sondaj kulesinin ana bileşenlerinden biridir. Drawworks kontrol etmek beceri ve deneyim gerektirir. Drawworks kontrol eden delici, sondaj kulesine yakın olduğu için çalışmak için çok zor bir konuma sahiptir. Bu nedenle, güvensizlik için şans daha fazladır. Drawworks sisteminin, herhangi bir hata oluşması durumunda delicinin ömrünü riske atabilir. Drawworks kontrol sistemini simüle ettim ve simülasyonu drawworks kontrol sisteminin prototip modeline uyguladım. Drawworks sistemi prototip modelinde delici ve drawworks güvenliği için deadman butonu, toprak hata dedektörü, ölü bant bölgesi eklendi. Ağırlık sondajda en önemli parametrelerden biridir. hareketli blok için ağırlık dengesi yapılmıştır, ağırlıkta herhangi bir değişiklik olursa, hareketli blok ağırlığa göre yukarı veya aşağı hareket edecektir. Böylece, bu güvenlik özellikleri uygulanarak, drawworks sistemi hata ve sarsıntılardan korunabilir, bu şekilde drawworks sistemi için olduğu kadar sondaj kulesine yakın çalışma konumu nedeniyle delici için güvenlik artırılır. Simülasyon ve uygulama için MATLAB ve Twincat3 yazılımı kullanılmıştır.

Anahtar Kelimeler: Drawworks, titreşimler, ağırlık dengesi, ölü bant bölgesi, Emniyet.

To My Parents, and Family

you will always be my source of strength

ACKNOWLEDGMENTS

I would like to take this opportunity to express my gratitude to Dr. Mehmet Efe Özbek, who has been a tremendous help to me during my thesis. During the project effort, I am grateful for his aspiring guiding, constructive criticism, and illuminating advise. Thank you also to my family for their unconditional support throughout this time. I must express my deepest gratitude to my family, without whom this achievement would not have been possible. Finally, God, my greatest supporter, I thank you.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

PI	Proportional Integral
PLC	Programmable logic controller
NC	Numerical Control
CNC	Computerized Numerical Control
BHA	Bottom whole assembly
WOB	Weight on the Bit

CHAPTER 1

INTRODUCTION

Due to fluctuation and uncertainty of crude oil prices, and its limited reserves make challenging task for energy policy makers to maintain its sufficient supply. Although increasing oil prices generally depend upon reserves, demand & supply and technology used for it to control drilling and refinery. So, use of advanced technology and control system in drilling has been recognized as one of main key factor in oil and gas field production. In oil and gas drilling industry, drawworks is one of main component that is used in connection with the derrick to hold, hoist and lower a drill string and its associated equipment attached with it in and out of the wellbore. Usually travelling block have an hook or other similar components that is used for the hosit and lower load attached with it. Hosit and lower operation of the travelling block is done by means of a hoist cable or line, one end of it is secured to the rig floor or ground is called "deadline", and other end of it is secured to the drawworks is called the "fastline". Drawworks is a rotatable cylindrical drum upon which the cable is wound by means of a suitable prime mover and power assembly [1]. Drawworks and traveling block assembly is controlled and operated by an operator is also called driller. Raising and lowering of the travelling block is controlled by the operator typically by way of a foot or hand throttle (joystick). During lower the travelling block, drawworks is supplied with brake. Primary brake, which is friction brake of either a band or disk type, is supplemented with emergency brake. Emergency brake is activated when the travelling block exceeds a maximum safe falling speed or in any emergency state. For hoisting steel cable is used here, other types of cable can also be used. During drilling cable string and associated equipment are very flexible and relatively easy to be damaged if any mishandling or fault occur even in a moderately deep well depth. Control the drill string and drill string weight applied to the drill bit is very important in drilling efficiency [2]. During drilling load on traveling block and assembly can exceed four hundred tons or more, minor operational error caused by the driller or a failure in any of the control system or rate of upward or downward movement of the traveling block could hazardous and even catastrophic, resulting in damage to equipment, personal injury, and even loss of human life.

Drilling operators are now more concerned about safety for control the drawworks [3]. Most offshore drilling rigs are equipped with the traditional brake as primary brake system and emergency brake as secondary. Brake is connected to the rotatable drum for control and limit the rotation of the rotatable drum and emergency stop. Drawworks drum is connected with electrical motor for rotation of the rotatable drum [4]. In this thesis simulation-based model and prototype model of drawworks has been observed at different speed. In prototype model weight sensor is connected with string it measures the load that is attached on the line. Weight equilibrium has been done on prototype model on 14 kg, if weight is changed travelling block would move upward or downward accordingly. Weight is very important parameter in drilling because it can disturb the whole system if it is beyond the capacity. So, weight equilibrium is very important for safe control of drawworks. Encoder position signal is connected to summation block with weight signal and analog signal output and their output is feed to PI controller here linear and non-linear both PI controllers has been used at different Kp and Ki values (tuning of controller), it gives required torque that is achieved at the end by TC module output. PI controller is used here to get steady state output, minimum overshoot, and rise & fall time of velocity. Drawworks drum rotates into two direction that is either forward or backward direction. So, when joystick position is in a backward direction, travelling block moves in upward direction, drawworks control system provides a signal for commencing a gradual release of the brake arrangement from the rotatable drum. When joystick position is in forward direction, travelling block moves in downward direction, drawworks control system gives signal to the brakes system to limit rotation of the rotatable drum. In prototype model of drawworks joystick gives a signal to the electrical motor for control the movement of the travelling block according to given signal. Here linear electrical output signal is provided by a movement control device that is throttle joystick. Safety features has been added to joystick by adding deadman button. Deadband region has been created in which joystick would not give an output signal when there is minor error or signal that is ±5% of neutral position of joystick. Ground fault detection has been added for more safety feature.

Simulation based model has been observed at different velocity, its rise & fall time and overshoot has been calculated by varying speed command. Simulation has been implemented to prototype model of drawworks by converting MATLAB Simulink to TwinCat3 using PLC module. Stateflow for joystick and motor has been simulated for prototype model of drawworks control system. In stateflow all the conditions have been mentioned for operating the drawworks. Safety features have been added in prototype model these are ground fault error detection, deadman button, deadman region and weight equilibrium. Results for prototype model of drawworks can be seen on MATLAB and TwinCat3 screen. Velocity has been calculated for drawworks system, its rise & fall time and overshoot has been calculated by using linear and non-liner PI controller on different Kp and Ki values. Methodology and results has been discussed in upcoming chapters and in the last conclusion.

1.1 Thesis Layout

There are five chapters in this thesis: Introduction, The Literature and Problem Statement, Methodology, Results, and Conclusion.

Chapter 2 (The Literature and Problem Statement) remotely control the drawworks, causes for failure of drawworks control system, important of different parameter used in drawworks and problem statement is discussed.

Chapter 3 (Methodology) construction and process has been discussed for drawworks control system.

Chapter 4 (Results) discussed the results and comments on each result.

Chapter 5 (Conclusion) discussed the conclusion of the research in details.

CHAPTER 2

THE LITERATURE AND PROBLEM STATEMENT

In research article [5], remotely control of drawworks and importance of safety at drilling site has been emphasized because driller work position is unsafe due to its working position near to drilling rig. In this article conventional drawworks system has been remotely controlled by joystick this way safety is increased due to remotely control.

In research paper [6], vibrations, its causes, types, and its drawbacks in drilling has been briefly described. Vibration occurs when frequency of the applied force and the natural vibration frequency of the drill string match. Drill string rotation at natural frequency causes vibration with severe tool damage including high shock loads, fatigue failure as washout and tool joint failure, etc. Axial, torsional, and transverse vibrations are the three different forms of vibration. Drilling can cause all three types of vibration. However, transverse vibration, which is more violent in vertical or low angle wells where drill string can move more freely than in vertical wells, is the most severe form. The price of drilling a well may rise by about 2–10% due to vibration-related issues, such as delays in pulling out the hole, and poor hole quality.

Research paper [7], describes the method for proportional integral (PI) controller tuning based on experimental data obtained from a test conducted in an open loop over the process to be controlled. PI parameters can be tuned using two different ways. The first type is the model-based approach, which relies on a model of the plant to perform accurate estimates of the controller parameters. However, it is going to be difficult to apply in real plants because it requires exact process model identification. The second type is the experimental tuning approach, which is less precise but much simpler to use because all that is required of the user is a simple experiment in the plant, take measurements in the response, and the apply some tuning equations with very little computational cost.

In research article [8], Optimal procedures for lowering and applying weight on the bit when starting to drill is briefly described. Initially bit is normally a few meters above the bottom when drilling operation is started. In order to prevent bit bouncing, continuous rotation must be achieved before the bit is placed into the bottom. Once the bit has reached at bottom, more weight is given to the bit by further lowering the drill string. This weight causes the vibrations and jerks if it is not applied properly during the drilling. Weight is also very helpful, when there is vibrations during drilling, driller adjusts parameters such as rotation speed or weight. As a result, the system's eigenfrequencies is changed, and vibrations is often damped.

Article [9], presents a method for calculating the weight on drill string. Initially weight has to be measured when it is in air. Then Buoyancy Factor (BF) is calculated, buoyancy factor is an upward force exerted by a fluid that opposes the weight of a partially or fully immersed object [10]. Then, multiply the weight in air by the buoyancy factor to determine the weight of the drill string in the drilling fluid. Mathematically,

Drill String Weight = Weight in air x Buoyancy Factor.

In patent [11], function of drive control for overload protection has been discussed. For drilling, motor is connected with drilling devices, drive control is arranged such that when a fraction of the motor torque that is available for the devices, when entire work range is reached, the complete mains voltage is applied. The motor torque/rotational speed curve follows the natural characteristic line until a load point where current limitation occurs, if still higher torque is required, due to it, operator of the drilling device receives a signal about the overload load condition in shape of noise that has created in the drilling device as the motor speed lowers because of normal work range is exceeded.

In this article [12], The objective is to analyze how a drill string's stability and vibration are impacted by drill string weight, bit weight, and angular velocity. In order to achieve this, the kinetic and potential energy of a rotating drill string are calculated while considering axial and lateral vibrations. Parameters for this model are weight of the drill string, the weight of the bit, and geometric shortening.

In research paper [13], Accuracy and correction for hook load measurement in drilling is described. During drilling operations, hook load is an important factor that is utilized to manage the weight on the bit and evaluate potential deteriorations of the downhole conditions. However, the hook load measurements accuracy is usually considered to be quite poor. In order to get higher-quality data, this research looks into the sources of uncertainty related to the hook load measurement and suggests correction techniques.

In this patent [14], method for troubleshooting of drilling well is described. The hook load is measured while free rotating, while the drill string is moved out of the well for an incremental distance, and during the drill string is moved into the hole for an incremental distance. These hook loads are used to calculate the drill string's effective friction factor as a function of depth. In order to identify potential hole issues, the hook loads and free rotating torque are plotted as a function of time.

From above research articles and patents, it is clear that safety is important factor during control of drawworks and weight plays an important role for that. So, in this thesis the main focus was to develop the control system of drawworks so that drilling should be done safely without any vibration and jerks also safety feature for driller has been focused. Vibration and jerks has been reduced using different ways and weight equilibrium has been added so that system should not failed due to overload.

CHAPTER 3

METHODOLOGY

3.1 Simulation and Implementation of drawworks control system

Simulation based model for control system of drawworks has been simulated. Also, it has been implemented on prototype model of drawworks for real time experience. Simulation based model have manual operating mode. MATLAB and Simulink have been used for simulation-based model of drawworks control system and for implement, simulation has been converted into TwinCat3 supported file. Main purpose for simulation-based model of a drawworks control system is to compare the difference between simulation-based model and prototype model of drawworks system and observe its results to get desire output. Implementation of simulation to prototype model of drawworks control system is done by using PLC and TwinCat3 software. Descriptions of software used in simulation and implementation are given below.

3.1.1 MATLAB

MATLAB programming language is a high-level programming language. It is one of useful software program for engineering and mathematics applications. MATLAB used for matrices, a mathematical chart, implementation of various algorithms, creation of graphical user interfaces, integration and conversion of programs to other programming languages. This software is widely used in both academia and industry. MATLAB was designed to manipulate data visualization tools at first, however there are now several toolboxes available for other applications [15].

3.1.2 Simulink

Simulink is a MATLAB-integrated model-based design environment. It is a graphical programming language tool that allows you to create block diagrams. Simulink environment supports features such as automatic code generation, modeling, and testing [16].

3.1.3 TwinCat3

The Windows Control and Automation Technology (TwinCAT) is a software system that makes practically any PC-based system into a real-time control system with various PLC, NC, and CNC. TwinCAT3 is the systematic development of TwinCAT2, and it is redefining the world of automation technology [17].

I have attached below diagram for simulation-based model and prototype model of drawworks control system.

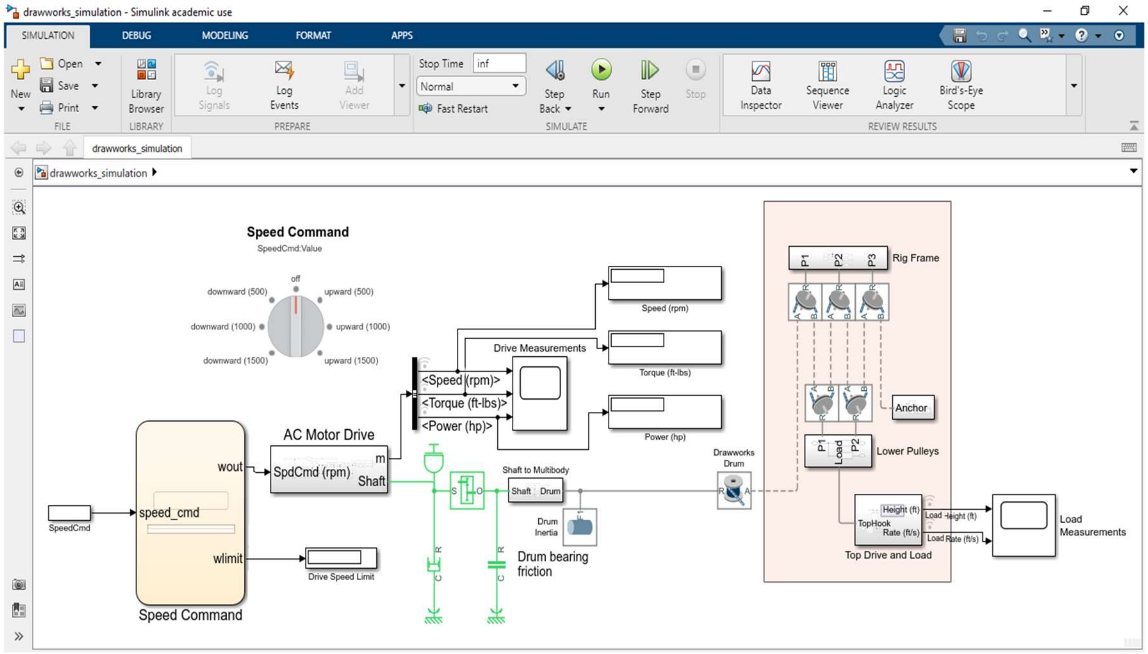


Figure 3.1 (a): Simulation based model of drawworks control system

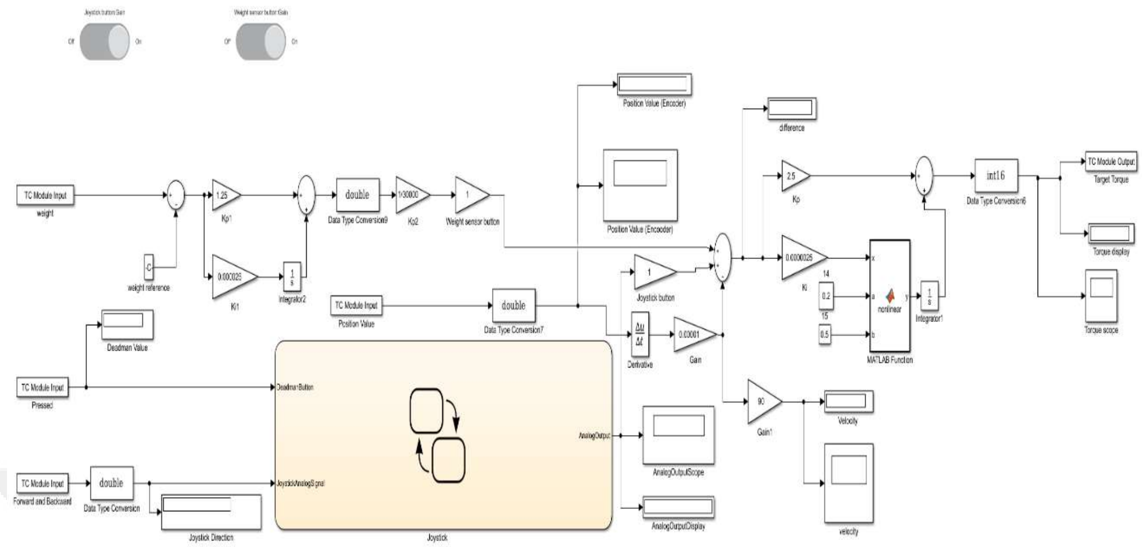


Figure 3.1 (b): Stateflow of Joystick for prototype model of drawworks

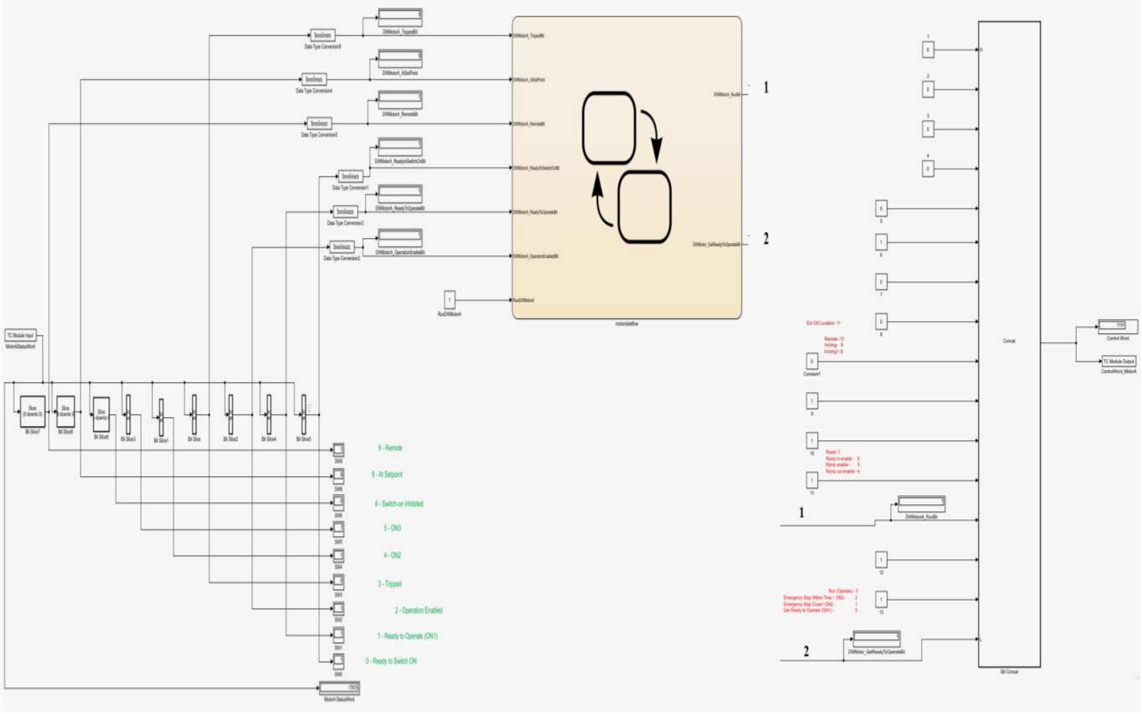


Figure 3.1 (c): Stateflow of Motor for prototype model of drawworks

3.2 Simulation based and prototype model of drawworks control system

In this section methodology and construction for simulation-based model and prototype model of drawworks control system has been discussed. First, I have discussed the methodology and construction for simulation-based model of drawworks control system then I have discussed prototype model of drawworks control system.

3.3 Simulation based model of drawworks control system

I have attached below diagram of simulation-based model for drawworks control system each section has been described in detail.

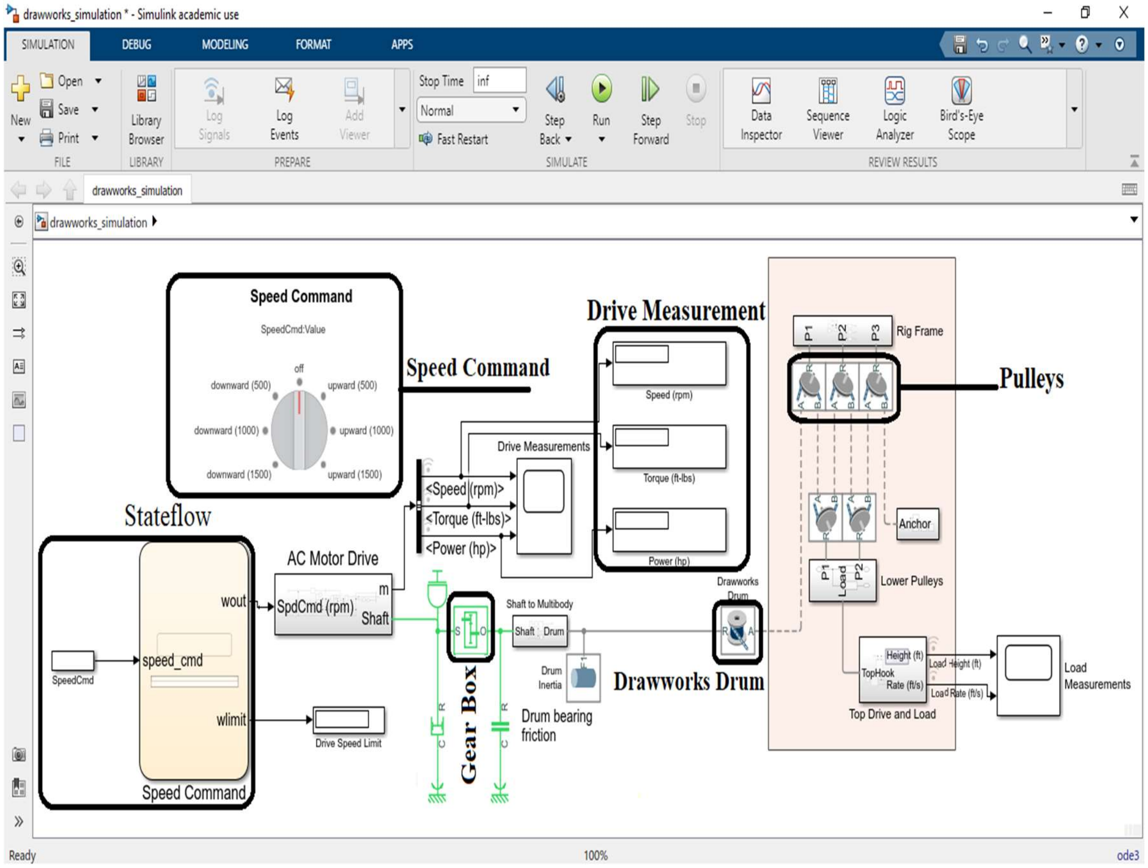
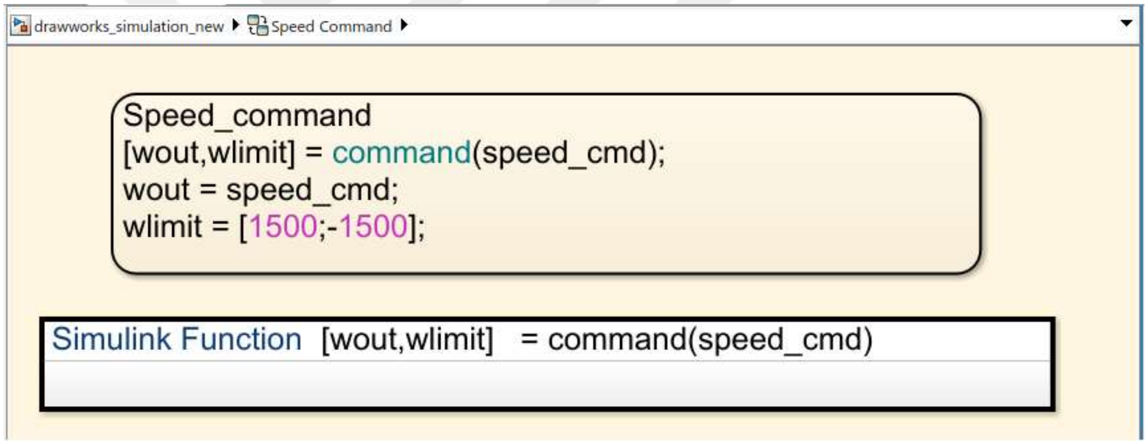


Figure 3.3 (a): Simulation based model of drawworks control system

1) Stateflow

Stateflow is a useful tool for creating and simulating models based on flow charts and state machines. The use of animation to demonstrate how the state machine works is one of the most important features of stateflow. It also performs static and run-time tests to observe whether the model can be executed or not. To create logic for input and output signals, events, and time-based conditions I have used the stateflow tool in this thesis. I have attached below the diagram of stateflow for simulation based model of drawworks control system.



```
Speed_command
[wout,wlimit] = command(speed_cmd);
wout = speed_cmd;
wlimit = [1500;-1500];
```

```
Simulink Function  [wout,wlimit]   = command(speed_cmd)
```

Figure 3.3 (b): Stateflow of simulation-based model of drawworks control system

In above stateflow all the terms and conditions are defined when it is in running mode it follows the defined conditions which are declared in stateflow, wout is speed output and wlimit speed limit. Outer diagram of above speed command stateflow is attached below.

Left side of port is the input port and right side of ports are output ports of speed command stateflow.

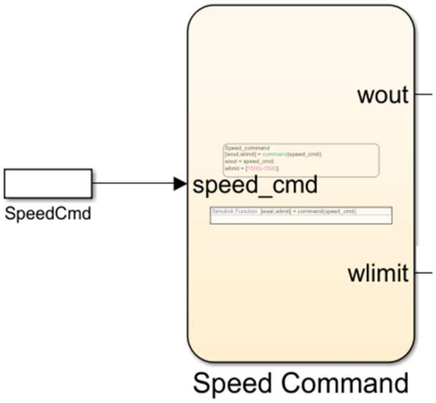


Figure 3.3 (c): Outer diagram of Speed Command stateflow

2) Speed Command

Operator changes the speed commands according to requirement so that desired output can be achieved. Speed command is given to motor, it runs according to given speed. Direction of travelling block is changed by speed command.

3) Drive Measurements

Drawworks parameter values and graph can be seen by drive measurement these are speed in rpm, torque in ft-lbs., and power in hp (horsepower). These three main parameters can be monitored and observed here by this drive measurement.

4) Gear Box

The Gear Box block represents a fixed gear ratio. The gear ratio is the ratio of the angular velocity of the input shaft to the angular velocity of the output shaft.

5) Drawworks Drum

Drawworks is the primary lifting equipment of a rotary drilling rigs. Its main task is to elevate and lower the travelling block. The drill string is moved up and down as the drum rotates because string is wound on the drawworks drum.

6) Pulley

A pulley is a wheel on an axle or shaft it allows a cable or line to move and change directions also transfer power between the shaft and the cable or line.

7) AC Motor Drive

An AC motor drive is a device that adjusts the speed of an electrical motor to improve control systems. It reduces energy consumption and increase energy efficiency. It also controls mechanical stress in motor.

3.3.1 Process and Working Principle for Simulation based model of Drawworks Control System

Initially speed command is set on which drawworks has to be run. When speed command is set and given to motor it starts to rotate according to it. Drawworks can rotate in two direction these are forward and backward direction. When drawworks rotate in forward direction travelling block move in upward direction and when drawworks rotate in backward direction travelling block moves in downward direction. When simulation is in running mode person should be alert when it reach its minimum and maximum height limit if travelling block direction is not changed when it reaches the maximum or minimum height limit it would give error and run mode would be stop. Minimum height for travelling block is -250 ft and maximum height is 50 ft (negative sign shows downward direction). Different parameters for motor can be calculated these are speed, torque, and power when drawworks is in running state. All the running conditions are set in speed command stateflow. Speed command stateflow is connected with AC Motor drive which control the speed of motor and drawworks drum. On drawworks drum cable is wound and it is connected with travelling block by pulleys. Pulleys support movement and change the direction of a travelling block and cable. When drawworks is in running mode rise & fall time and overshoot of speed parameter is measured that is our main concerned.

3.3.2 Flow chart for Simulation based model of drawworks control system

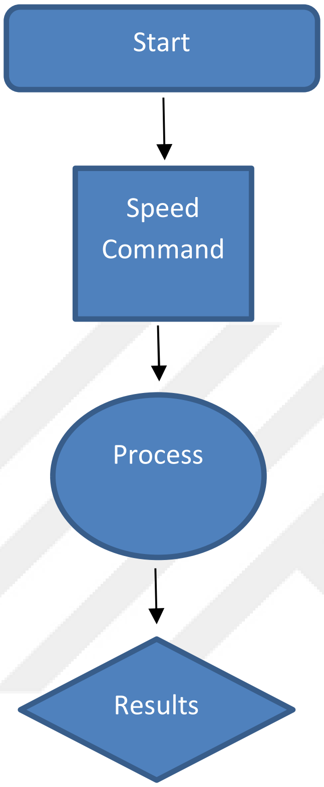


Figure 3.3.2: Flow chart for Simulation based model of drawworks control system

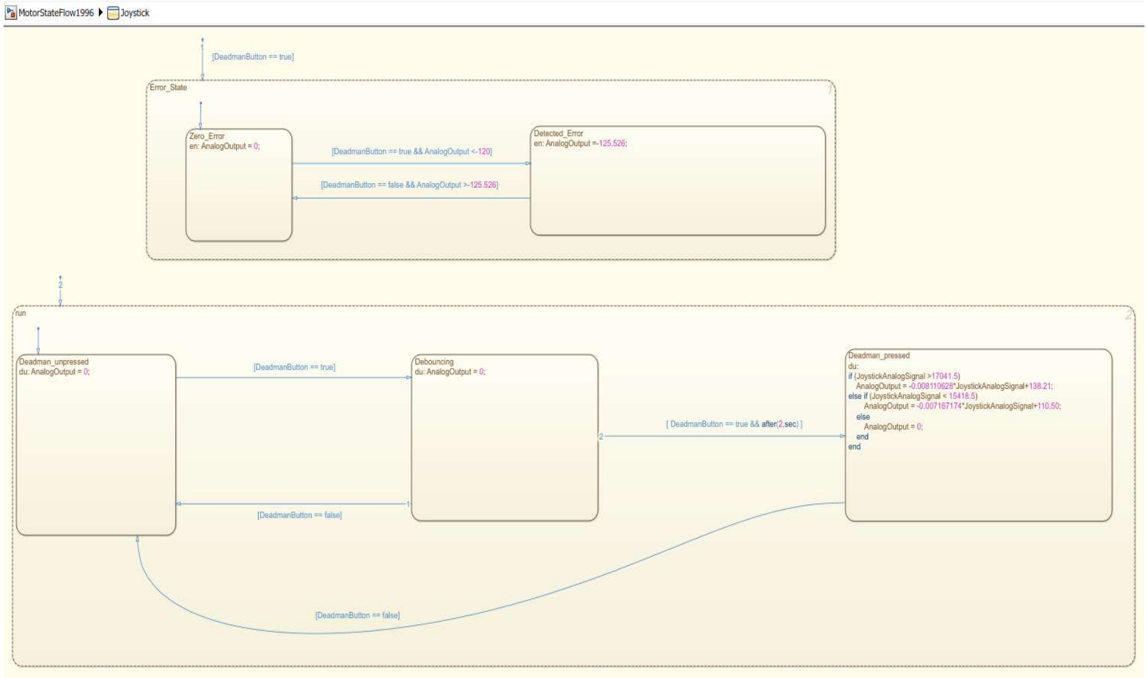
Initially speed command is given to the system and drawworks act and run according to given speed command. After that it process the command and travelling block move in upward or in downward direction accordingly in the end, desired output is achieved.

3.4 Prototype model of drawworks control system

For prototype model of drawworks control system, I have designed stateflow for joystick and motor. As drawworks is controlled by joystick and it is rotated by motor.

3.4.1 Working principle for prototype model of drawworks control system

Simulation-based model for control system of drawworks has been observed. Now in this section prototype model of drawworks control system would be discussed. Prototype model of drawworks is controlled by joystick and rotate by motor. So, to control drawworks system two stateflow are required these are joystick and motor. Stateflow has been already discussed so here I would discussed more about state and transition condition. Let's start with stateflow of joystick.



```
du:
if (JoystickAnalogSignal >17041.5)
   AnalogOutput = -0.008110628*JoystickAnalogSignal+138.21;
else if (JoystickAnalogSignal < 15418.5)
      AnalogOutput = -0.007167174*JoystickAnalogSignal+110.50;
   else
      AnalogOutput = 0;
   end
end
```

Figure 3.4.1 (a): Stateflow of Joystick for Prototype model of drawworks

From above stateflow of joystick it can be observed that when deadman button is pressed initially error would be checked that is ground fault, if there is any error detected it would not processed further until the error is removed. It protects the system from ground fault. When system is not grounded properly it always produced some backward direction signal at analog output. So, for protection at initial stage it checks either system has some error or not. If system has error, it would not processed further. Transition conditions for checking ground fault error are deadman button and analog output signal. When deadman button is pressed it gives signal at analog output, if there is an error the analog signal would be <-120 and if there is not error analog output signal would be >-125.526. After checking Error_State it goes into run state. Deadman button act as safety button in joystick without it joystick would not give an output signal. Run state is divided into three state these are Deadman_unpressed, Debouncing and Deadman_pressed. Initially run state is changed from deadman_unpressed to debouncing state when deadman button is pressed. If still deadman button is pressed and 2 seconds (delay) is passed it would go further for next state that is deadman_pressed state and follow the condition which are defined in stateflow. If deadman button is unpressed at any state it would go back to initial state of run. To avoid jerks and produce smooth grow of analog output signal, conditions have been defined at the final state of run. Values has been calculated by neutral position of joystick ±5% that region is called deadband region here joystick would not give output signal. Slop has been used here for gradual increase of output signal and safety feature, if there is any small error signal in between deadband region joystick would not give an analog output. I have attached below the diagram of deadband region of joystick.

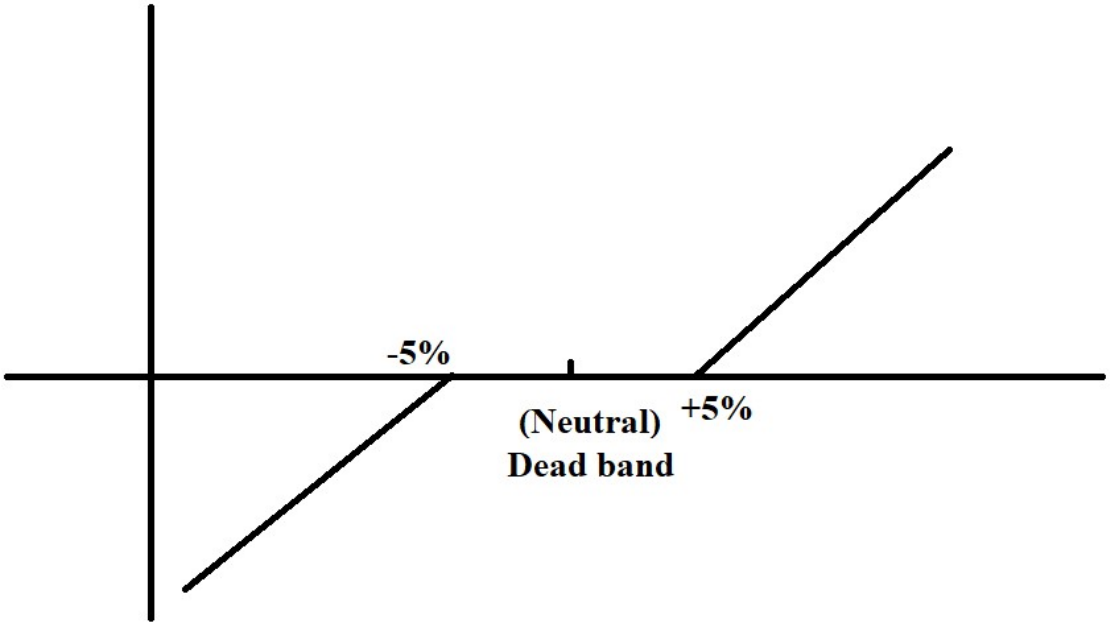


Figure 3.4.1 (b): Deadband region of Joystick

Outer diagram for stateflow of joystick has been attached below. There are two inputs for joystick these are DeadmanButton and JoystickAnalogSignal and one output signal that is AnalogOutput. Analog output is connected with Position Value (Encoder) and weight sensor signal by summation block their output is connected to PI controller. Linear and nonlinear PI controller are used individually to get desired output. After tune of PI controller its output is converted into int16 form by using convertor because system requires that form for torque signal at the end desire output is achieved by TC Module Output.

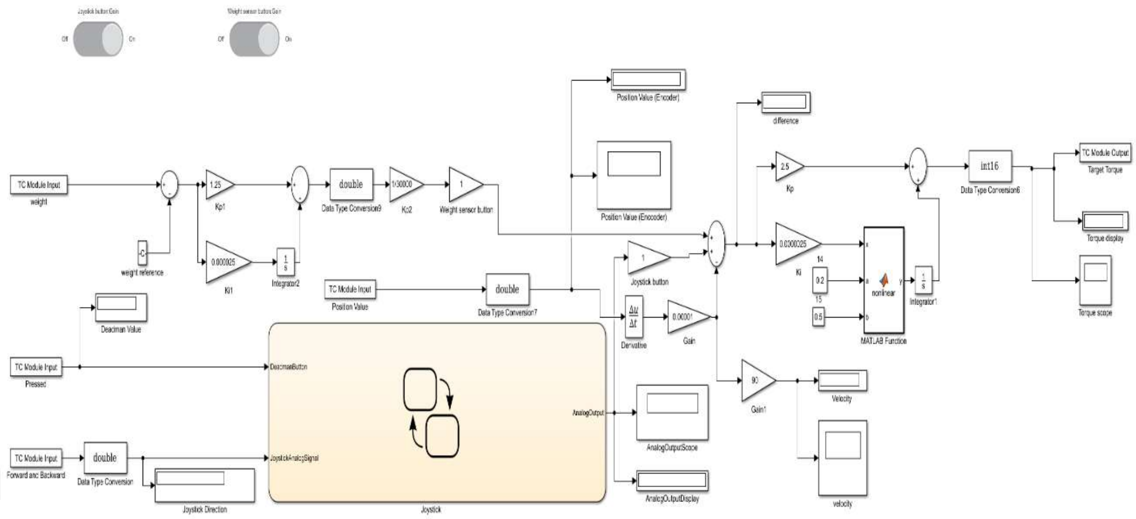


Figure 3.4.1 (c): Stateflow of Joystick connected with different equipment

Now Stateflow of motor would be discussed. I have attached below the stateflow diagram of drawworks motor. It has been divided into two states these are InitializeAndRunState and StopState. InitializeAndRunState start by entering into IntState here it checks the transition condition that is RunDWMotorA if it is true, it would further processed otherwise it would go to StopState. If RunDWMotorA is true and DWMotorA_RemoteBit is also true then it would go into next stage of state block. Here motor run_bit and GetreadytoOperateBit conditions are performed at entry after this it moves further for next state by following transition conditions that are 500-msec of delay and DWMotorA_ReadyToOperateBit when entering this state DWMotorA_RunBit=true is performed at entry level. After that again transition for next state is checked at this stage DWMotorA_OperationEnabled is check if it is true it goes further for next stage by following transition condition of 500-msec delay and DWMotorA_AtSetPoint here at this state motor start to operate and drawworks began to rotate if its set point is true. So, this stateflow continuously work like this cycle. Stateflow for motor is attached below all these conditions can be observed from that stateflow diagram.

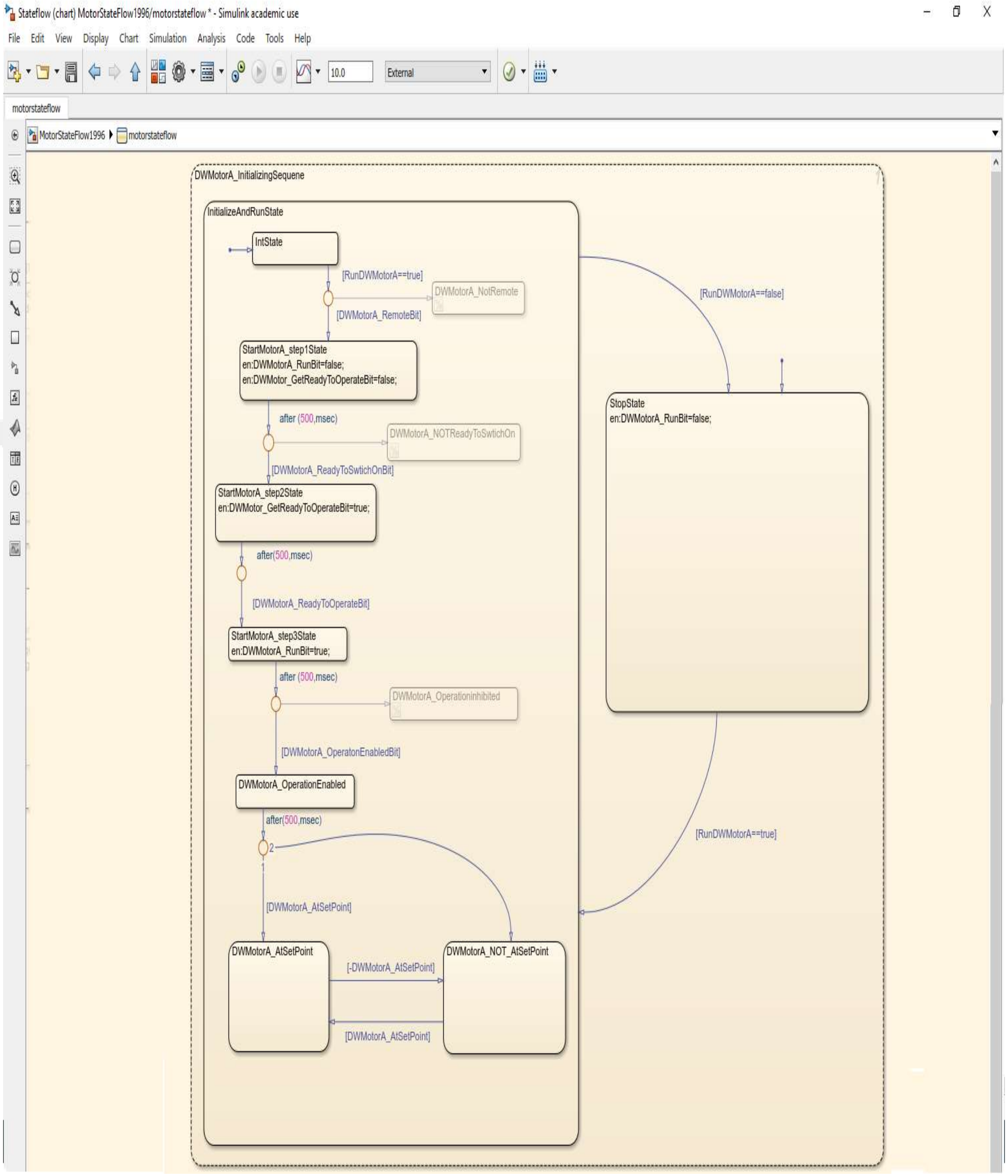


Figure 3.4.1 (d): Stateflow for motor

Stateflow of motor has seven inputs and two output. These input and output ports are connected with different components these are TC Module input, Bit Slice, converter, constant and displays. Outer diagram for motor stateflow has been attached below along with attached components.

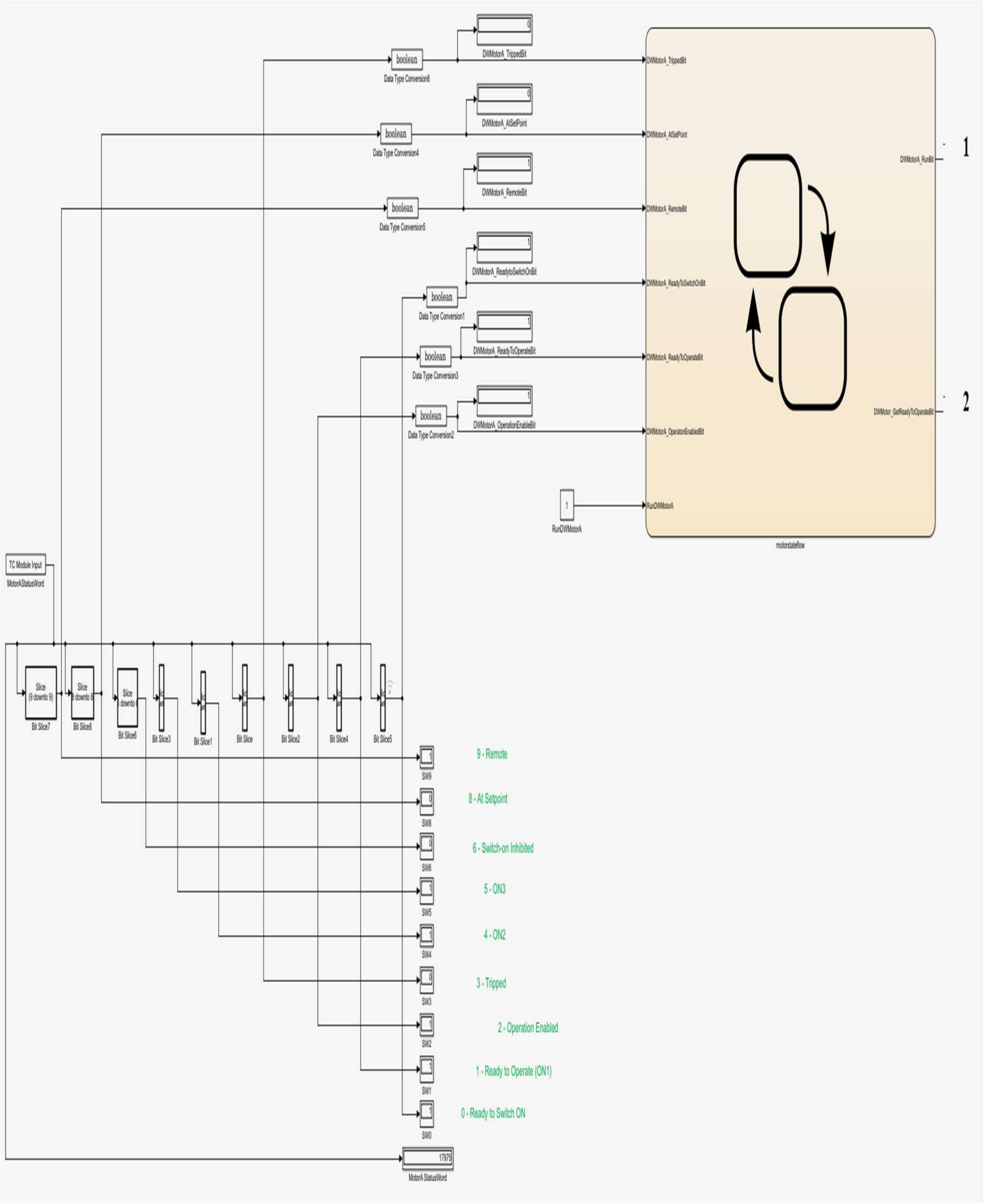


Figure 3.4.1 (e): Stateflow of motor connected with different equipment (a)

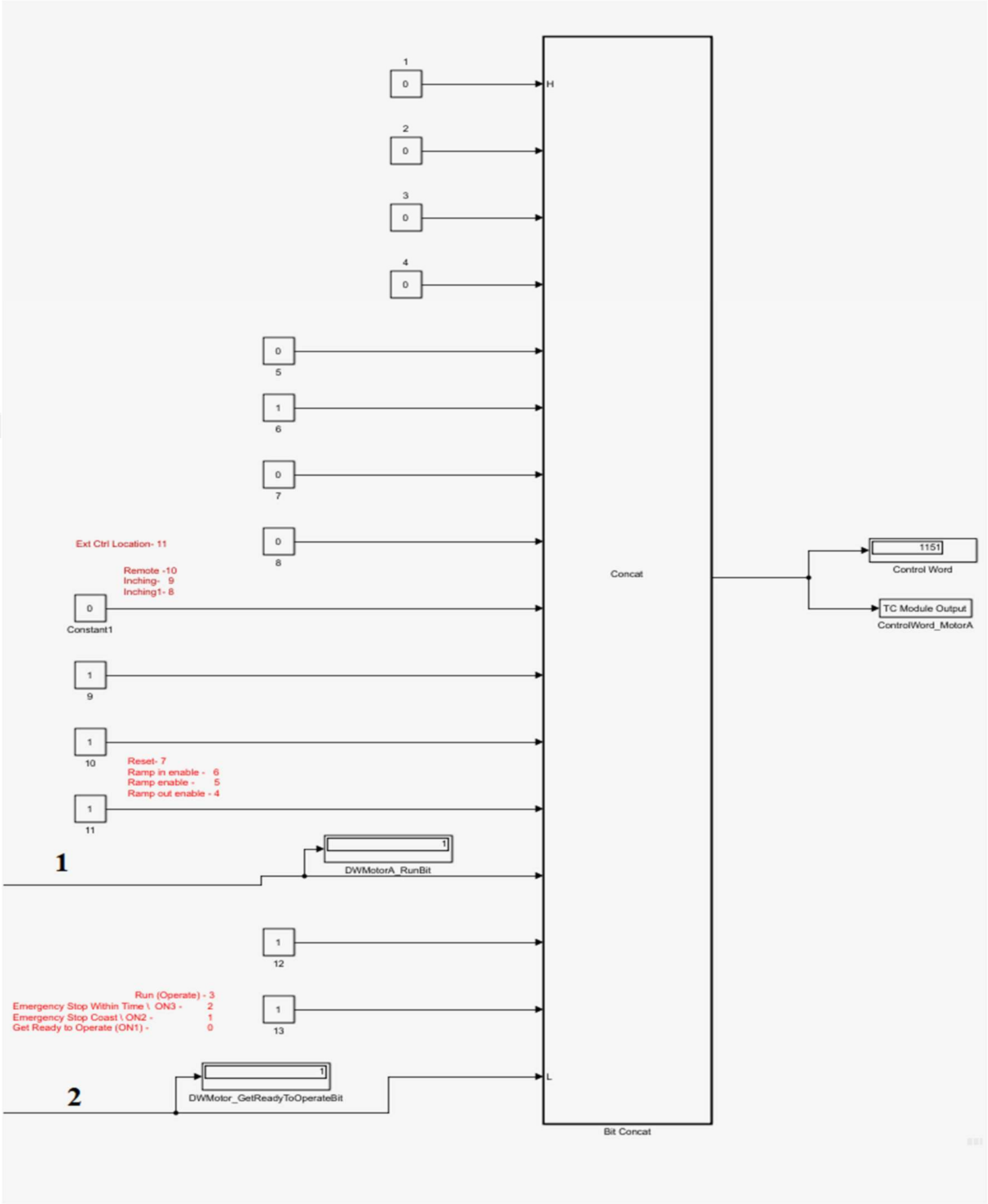


Figure 3.4.1 (f): Stateflow of motor connected with different equipment (b)

3.4.2 Process for Integration of Matlab with TwinCat3

There is lengthy process to integrate the MATLAB Simulink with TwinCat3. For that it must follow below mentioned steps these are as follows.

1) Download and install one of the supported Visual Studio versions. Pay special attention to the C++ components' installation.

2) Then install TwinCAT3 and integrate it with the installed version of Visual Studio.

3) Install the Microsoft Windows Driver Kit.

4) Install TE1400 in the TwinCAT folder.

Restart the target system. When all these steps are followed successfully, Test mode text will appear at the bottom of desktop.

5) Open MATLAB as administrator and execute.

C:\TwinCAT\Functions\TE1400TargetForMatlabSimulink\SetupTwinCatTarget.pin MATLAB it can be seen from below diagram.

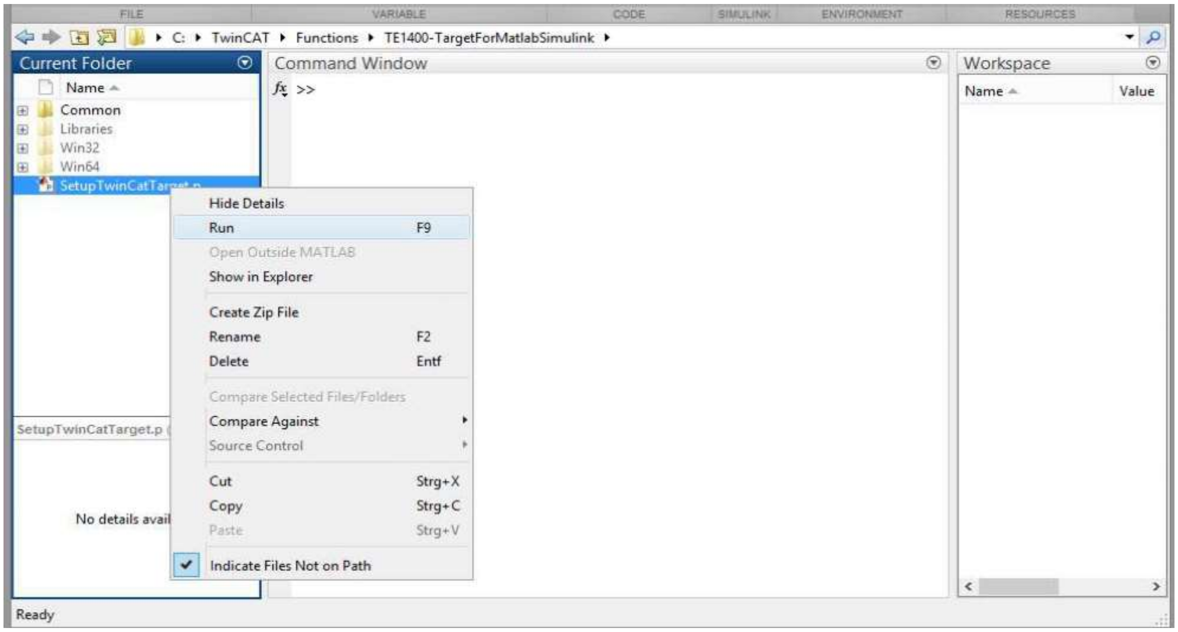
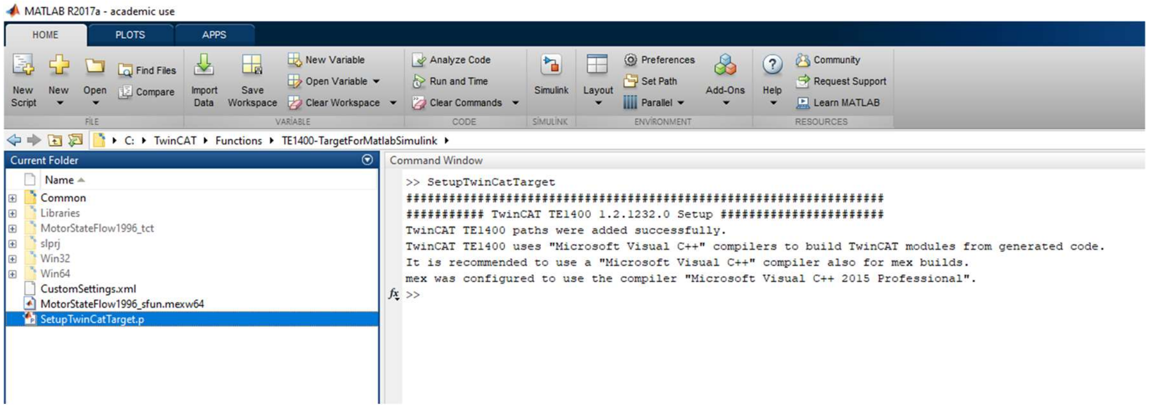


Figure 3.4.2 (a): TE1400 File installation for MATLAB Simulink

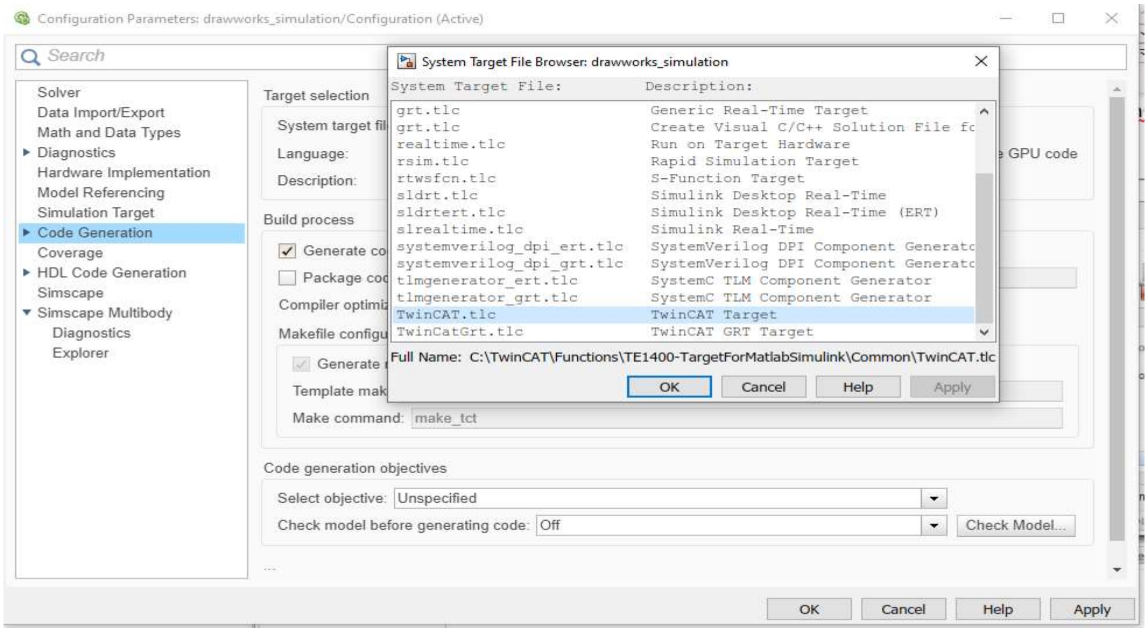


```
>> SetupTwinCatTarget
###################################################################
########### TwinCAT TE1400 1.2.1232.0 Setup #######################
TwinCAT TE1400 paths were added successfully.
TwinCAT TE1400 uses "Microsoft Visual C++" compilers to build TwinCAT modules from generated code.
It is recommended to use a "Microsoft Visual C++" compiler also for mex builds.
mex was configured to use the compiler "Microsoft Visual C++ 2015 Professional".
fx >>
```

Figure 3.4.2 (b): TE1400 File after installation for MATLAB Simulink

6) Change the address of MATLAB to C:\TwinCAT\3.1\Config\Modules because TE1400 plugin puts TcCOM objects in that folder.

Above mentioned steps must be followed if any of step is missed MATLAB would not generate supported file for TwinCat3. For code generation of desired file open Model Settings shortcut key for that is Ctrl+E open Code Generation menu then select TwinCAT.tlc as "System target file".



☑ Generate co

Make command: make_tct

System Target File:	Description:
grt.tlc	Generic Real-Time Target
grt.tlc	Create Visual C/C++ Solution File fc
realtime.tlc	Run on Target Hardware
rsim.tlc	Rapid Simulation Target
rtwsfcn.tlc	S-Function Target
sldrt.tlc	Simulink Desktop Real-Time
sldrtert.tlc	Simulink Desktop Real-Time (ERT)
slrealtime.tlc	Simulink Real-Time
systemverilog_dpi_ert.tlc	SystemVerilog DPI Component Generatc
systemverilog_dpi_grt.tlc	SystemVerilog DPI Component Generatc
tlmgenerator_ert.tlc	SystemC TLM Component Generator
tlmgenerator_grt.tlc	SystemC TLM Component Generator
TwinCAT.tlc	TwinCAT Target
TwinCatGrt.tlc	TwinCAT GRT Target

Full Name: C:\TwinCAT\Functions\TE1400-TargetForMatlabSimulink\Common\TwinCAT.tlc

Select objective: Unspecified

Check model before generating code: Off

Figure 3.4.2 (c): Convert Simulink file into TwinCat3

In addition, a fixed-step solver must be configured in the solver settings, to ensure real-time capability of the simulink model. Click on apply and ok button after these changes generate the code by using Code generator when code is successfully generated successful notification is appeared on screen. After this notification tmc file can be seen in TwinCat3 for that double click at TcCOM Objects section and add it. Now file is ready for TwinCat3 and can be implemented to targeted system using PLC. For integration of MATLAB Simulink with TwinCat3 TC1000, TC1300, TC1320 and TE1400 license must be installed without these licenses desired simulation cannot not be implemented on targeted system.

3.4.3 Construction of prototype model of drawworks.

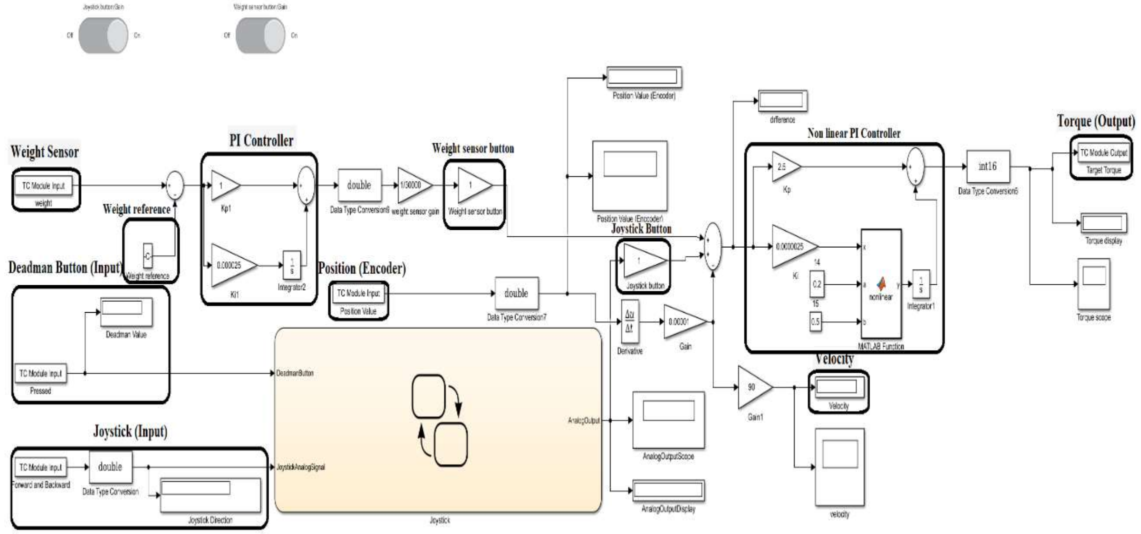


Figure 3.4.3 (a): Simulation for prototype model of drawworks using nonlinear PI controller

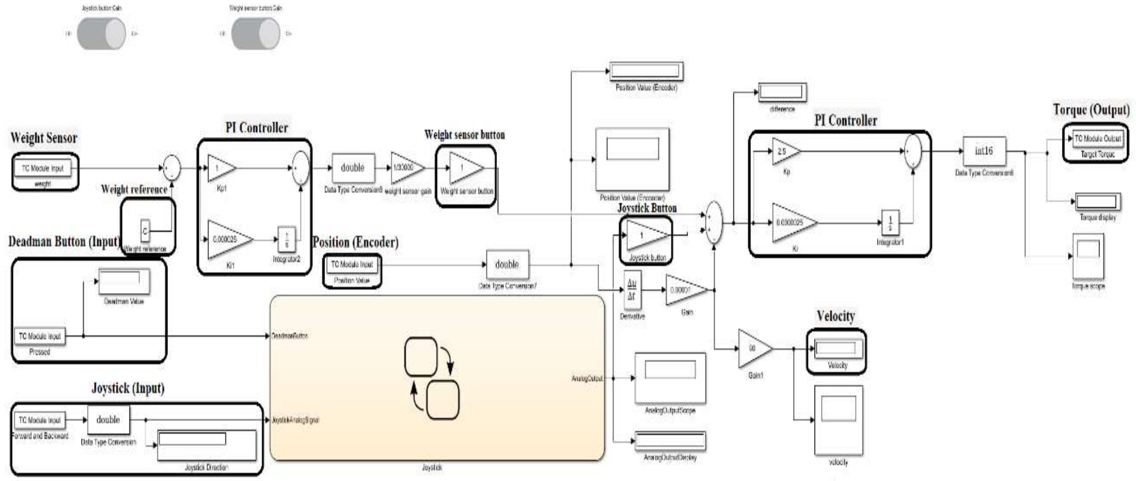


Figure 3.4.3 (b): Simulation for prototype model of drawworks using linear PI controller

It can be seen from above diagram that joystick stateflow is connected with different components these are Tc Module input, converter, PI Controller that is either linear or linear, gain, display, scope, and TC Module Output. There are two TC Module input attached with stateflow of joystick and one Analog output. I have attached below the diagram of joystick that has been used in prototype model of drawworks.

Figure 3.4.3 (c): Joystick used in Prototype model

JoystickAnalogSignal (Input) is connected with convertor that convert signal into double form. Analog Output signal is connected by summation block with encoder position value and weight sensor output, their error difference is feed into PI controller. Linear and non-linear PI controller has been used individually at different Kp and Ki values. Kp is proportional gain and Ki is integral gain. By varying values of Kp and Ki steady state error is minimized, minimum overshoot and lower rise & fall time is achieved. Vary the Kp and Ki values to get desired output is basically tuning of PI controller. When nonlinear PI controller is used Ki value is connected with MATLAB function block in which nonlinear equation is described. When linear PI controller is used there is not any Matlab function block. PI controller here helps to reduce the overshoot and rise & fall time of velocity wave which is beneficial for our motor used in our system that rotates the drawworks. Position encoder is attached with derivate and gain block, its output is velocity of prototype model. Output of PI controller is feed into convertor that convert signal into required output form that is int16 at the end required targeted torque is achieved by TC Module Output.

3.4.4 Weight Scaling

Prototype model has weight sensor which measured the weight that is attached with string cable. Weight sensor gives value in int32 format. I have measured the weight starting from zero to increase one by one by adding 7 kg extra weight until 28 kg weight is added and then scaled it. Graph and table for scaling of weight is attached below.

Table 3.4.4: Weight Scaling Table

Weight(kg)	Weight Value in INT32 (by weight sensor)
0	-10575792
7	-4752896
14	-294400
21	4182016
28	8299008

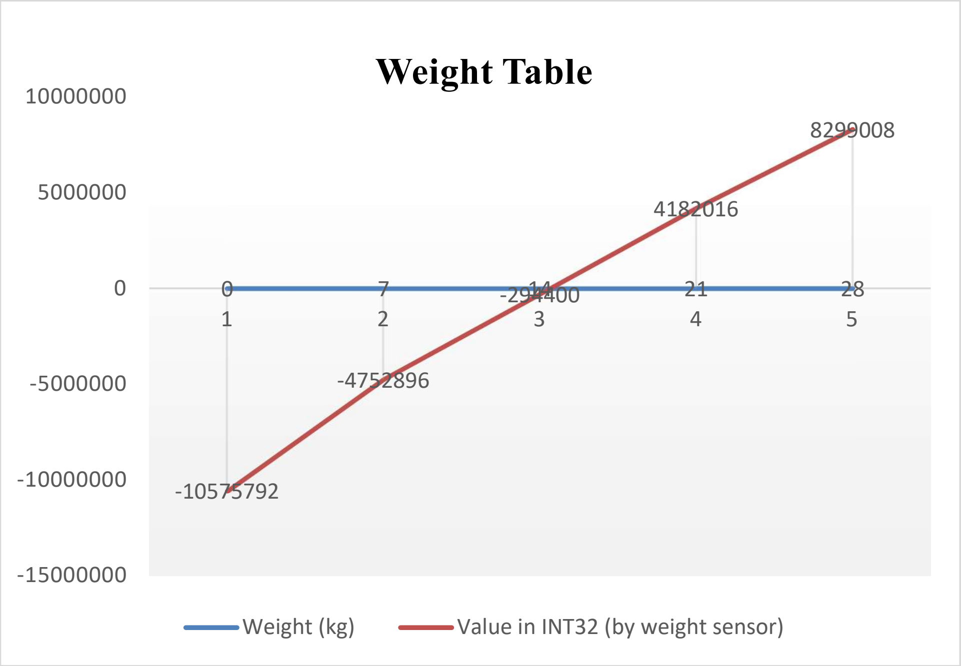


Figure 3.4.4: Weight Scaling Graph

3.4.5 Weight Equilibrium

Weight sensor is attached with string which measure the load attached with it. Weight sensors act an important role in drilling. Because it aware about capacity of motor when it rotates the drawworks. If weight is beyond the capacity of motor vibrations and noise occurred. So, weight readings alerts the driller before any fault occurred due to it. Here travelling block has been weight equilibrium on 14 Kg. It can be seen from simulation; weight sensor is connected with reference weight that is 14 kg by summation block and their output is connected with PI controller. Here PI controller is used to get steady state error output. Output of controller is connected with gain and switch so that weight equilibrium can be turn on and turned off by using switch according to the requirement. Output of PI controller is feed into input of other PI controller by simulation block. Here position encoder value, analog output value and weight equilibrium are connected by summation block. When weight is added more than the 14 kg travelling block goes downward as weight sensor measure the weight and give signal to run the motor so that travelling block move downward. On the other hands when weight is lower than 14 kg, weight equilibrium is again disturbed and travelling block move in upward direction by giving signal to motor so that travelling block move. When drawworks system has not been operated at that time if there is change in weight, weight sensor would give a signal to motor for move the travelling block depend upon the weight accordingly. So, weight parameter can drive the drawworks if weight equilibrium is disturbed. Weight reference can be changed according to the requirement it is basically the value on which travelling block would be in equilibrium state. Weight equilibrium is very helpful because it alerts when weight is changed either greater than or less than the reference weight. By this way system can be prevent from the fault. If there is any change in weight it causes the vibrations and noise which can create fault. So, by weight equilibrium these faults can be prevent. Switch used here in weight equilibrium is to connect and disconnect the weight equilibrium according to the requirement. I have attached below diagram which demonstrates the weight equilibrium.

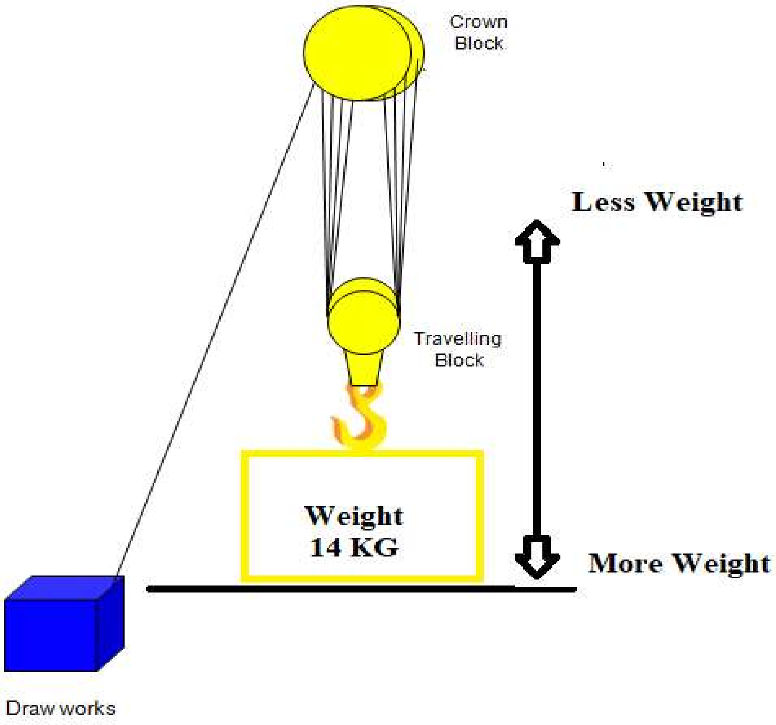


Figure 3.4.5: Weight Equilibrium

From above diagram, arrow shows the direction of movement for travelling block. If weight is less than reference weight travelling block moves in upward direction, if weight is more than reference weight travelling block would move in downward direction.

3.4.6 Components used in prototype model of drawworks control system

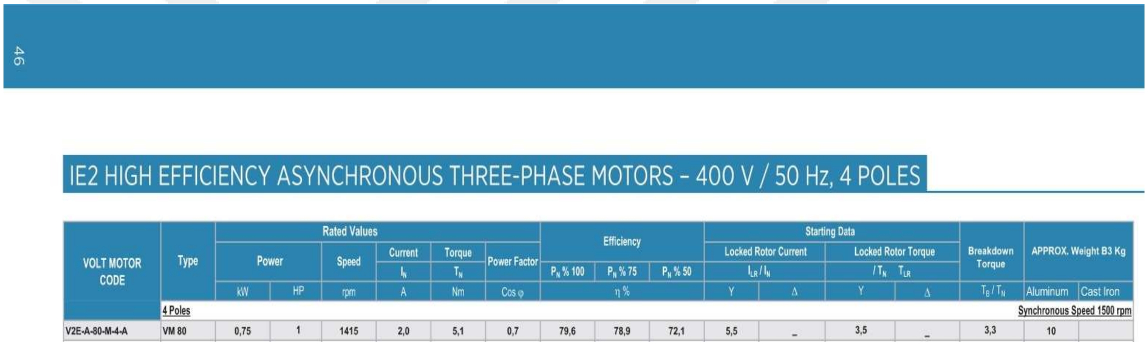
Real drawworks control system might be changed if it is compared with prototype model of drawworks because in prototype model environment is in control, as it is performed in lab while in real system it is not. So, there is always difference in prototype model results and real model results. I have attached below prototype model of drawworks control system.

Figure 3.4.6 (a): Prototype model of drawworks control system

Components used in this prototype model are actuator, drawworks drum, weight sensor, embedded PC, motor driver, joystick, and emergency button. All these components would be discussed one by one in details.

a) Actuator

The actuator is manufactured by VOLT Electric motor is AC (Alternating Current) 3-phase asynchronous electric motor its type is V2E-A-80-M-4-A with 400V voltage, 50 Hz frequency, 2 A current, and 0.75 KW power. The maximum rotational velocity is 1415 rpm, and the nominal torque is 5.1 Nm [18]. I have attached below the datasheet and diagram of actuator below. The driver can provide the motor with rotational speed and torque reference signals. On the electric motor reduction gearbox is attached it is manufactured by Hidesan 1400/32 rpm input to output rotational velocity ratio.



46

IE2 HIGH EFFICIENCY ASYNCHRONOUS THREE-PHASE MOTORS – 400 V / 50 Hz, 4 POLES

VOLT MOTOR CODE	Type	Rated Values						Efficiency			Starting Data				Breakdown Torque	APPROX. Weight B3 Kg	
		Power		Speed	Current	Torque	Power Factor				Locked Rotor Current		Locked Rotor Torque				
					I_N	T_N		P_N % 100	P_N % 75	P_N % 50	I_{LR} / I_N		$/ T_N$ T_{LR}				
		kW	HP	rpm	A	Nm	Cos φ	η %			Y	Δ	Y	Δ	T_B / T_N	Aluminum	Cast Iron
	4 Poles																Synchronous Speed 1500 rpm
V2E-A-80-M-4-A	VM 80	0,75	1	1415	2,0	5,1	0,7	79,6	78,9	72,1	5,5	_	3,5	_	3,3	10	

Figure 3.4.6 (b): Data Sheet of AC motor type V2E-A-80-M-4-A

Figure 3.4.6 (c): AC motor type V2E-A-80-M-4-A

b) Drawworks drum

The drawworks drum is one of main part of drilling rigs. It is used for wounding a hoisting line or cable on that drum. I have attached below diagram of drawworks drum used in my prototype model.

Figure 3.4.6 (d): Drawworks drum

c) Weight Sensor

A weight sensor manufactured by ESIT, type TB1000, is used in prototype model. It measures the hook weight and tension force along the deadline. It has a capacity of 1000 kg, with a minimum load of 200 g and a maximum overload capacity of 1500 kg [19]. I have attached below the picture of weight sensor used in my prototype model.

Figure 3.4.6 (e): Weight Sensor

d) Embedded PC

I have used a CX series Embedded PC unit in my drawworks prototype model, it has numerous I/O modules to construct a space-saving industrial controller. Beckhoff Company have manufactured it. Its type is CX2100. It has a 2x16 character FSTN LC display, a selector switch, and an enter key for querying internal status values as well as self-created display parameters or menu with and without input options [20]. The operating system is Microsoft Windows Embedded Compact, and the control software is TwinCAT 3 runtime. I have attached below the diagram of CX2100.

Figure 3.4.6 (f): CX2100 Embedded PC

e) Motor Driver

ABB's type ACS880 motor drive has been used. It can control a wide range of applications. Main features of this motor drive is it can direct torque control (DTC) for precise open and closed loop. It has also an energy optimizer and data of energy efficiency for monitoring and saving electricity [21]. I have attached below diagram of our system motor drive.

Figure 3.4.6 (g): ABB's type ACS880 motor drive

f) Joystick

Dual axis industrial joystick model JC80 manufactured by Avioni company has been used. It has following specifications [22].

Table 3.4.6: Joystick Specification

Operational Voltage range	DC5V/DC12V/DC24V
Electrical Angle	36.67°
Resistance Range	1.8±20%KΩ ; 2±20%KΩ ; 2.9±20%KΩ; 5±20%KΩ
Accuracy	≤ %±2
Protection Class	IP65

I have attached below the diagram of Joystick that is used in prototype model.

Figure 3.4.6 (h): Joystick Model JC80

CHAPTER 4

RESULTS

In this chapter results and graphs have been discussed for simulation based and prototype based model of drawworks control system. Results and graphs varied according to different parameters and in the end comments are written on each results according to our desired output.

4.1 Results of simulation-based model of drawworks control system

Travelling block has been observed while moving upward and downward direction on different speed. Speed rise & fall time and overshoot has been observed.

a) Travelling block moving upward direction

Travelling block would be operated at different speed for observe the rise time and overshoot of velocity.

1) Travelling block moving upward direction at 1500 rpm

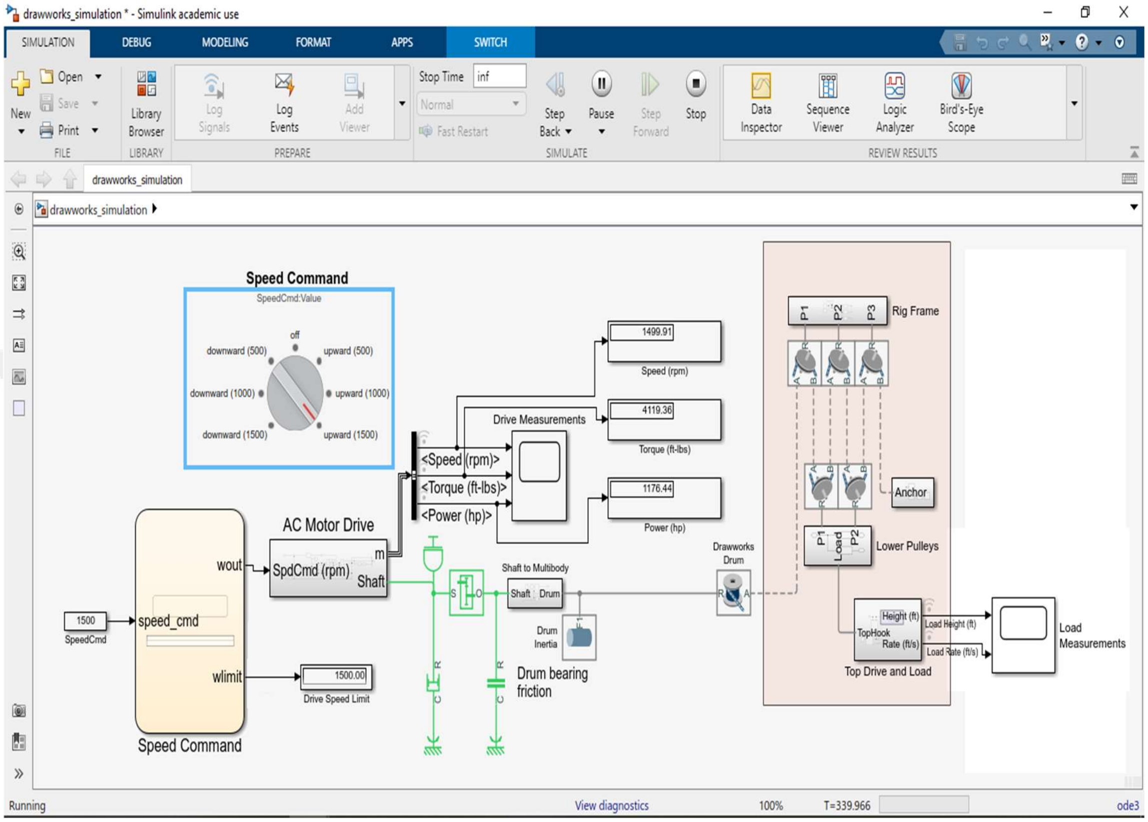


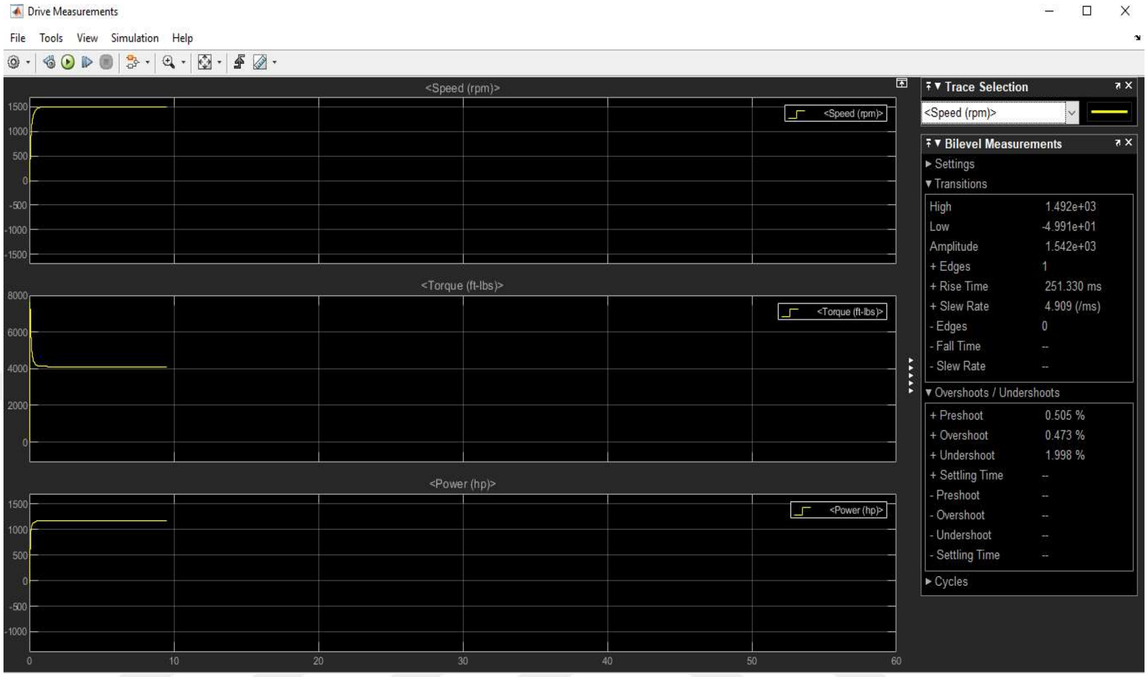
Figure 4.1 (a): Travelling block moving upward direction at 1500 rpm

Travelling block is moving in upward direction. Speed has been set at 1500 rpm. Torque is 4119.36 ft-lbs and Power is 1176.44 hp.

Table 4.1 (a): Travelling block moving upward direction at 1500 rpm

Speed (rpm)	1500
Torque (ft-lbs)	4119.36
Power (hp)	1176.44

I have attached below graph for speed, torque and power. In that graph rise time and overshoot for velocity has been measured.



<Speed (rpm)>

High	1.492e+03
Low	-4.991e+01
Amplitude	1.542e+03
+ Edges	1
+ Rise Time	251.330 ms
+ Slew Rate	4.909 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--
+ Preshoot	0.505 %
+ Overshoot	0.473 %
+ Undershoot	1.998 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

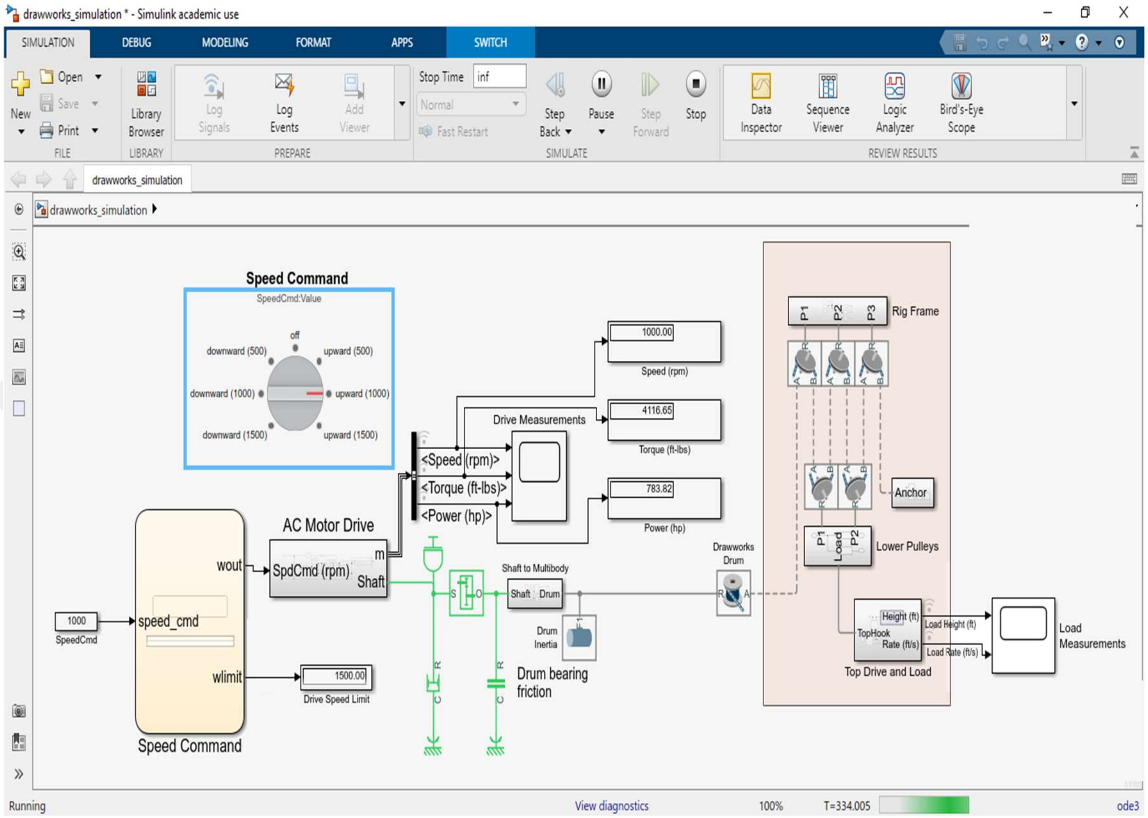
Figure 4.1 (b): Graph for speed, torque, and power at 1500 rpm

Table 4.1 (b): Rise time and overshoot at 1500 rpm

Rise Time (ms)	251.330
Overshoot (%)	0.473

Rise Time for speed at 1500 rpm is 251.330 ms and overshoot is 0.473% when travelling block is move upward direction.

2) Travelling block moving upward direction at 1000 rpm



Stop Time: inf

Speed (rpm): 1000.00

Torque (ft-lbs): 4116.65

Power (hp): 783.82

Drive Speed Limit: 1500.00

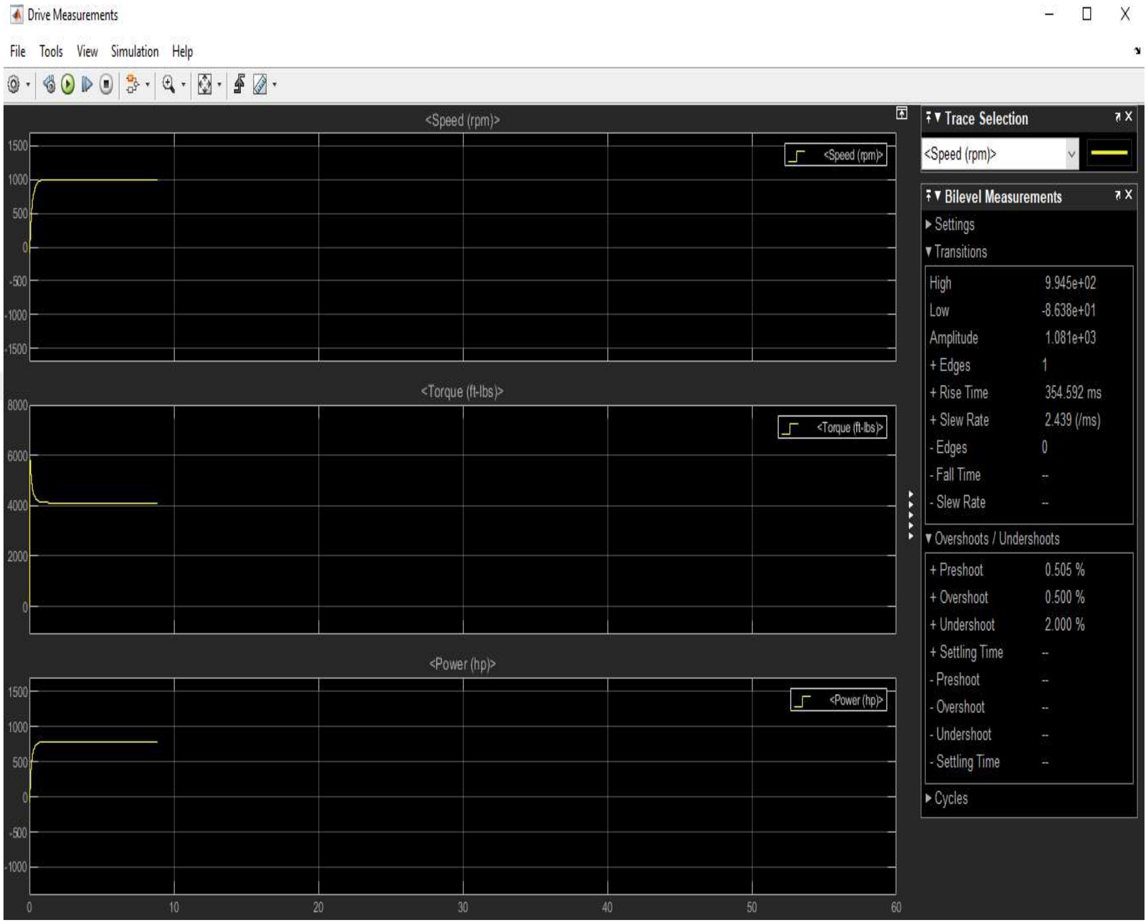
Figure 4.1 (c): Travelling block moving upward direction at 1000 rpm

Travelling block is moving in upward direction. Speed has been set at 1000 rpm. Torque is 4116.65 ft-lbs and Power is 783.82 hp.

Table 4.1 (c): Travelling block moving upward direction at 1000 rpm

Speed (rpm)	1000
Torque (ft-lbs)	4116.65
Power (hp)	783.82

I have attached below graph for speed, torque and power. In that graph rise time and overshoot for velocity has been measured.



Trace Selection: <Speed (rpm)>

High	9.945e+02
Low	-8.638e+01
Amplitude	1.081e+03
+ Edges	1
+ Rise Time	354.592 ms
+ Slew Rate	2.439 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--
+ Preshoot	0.505 %
+ Overshoot	0.500 %
+ Undershoot	2.000 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Figure 4.1 (d): Graph for speed, torque, and power at 1000 rpm

Table 4.1 (d): Rise time and overshoot at 1000 rpm

Rise Time (ms)	354.592
Overshoot (%)	0.5

Rise Time for speed at 1000 rpm is 354.592 ms and overshoot is 0.5% when travelling block is move upward direction.

3) Travelling block moving upward direction at 500 rpm

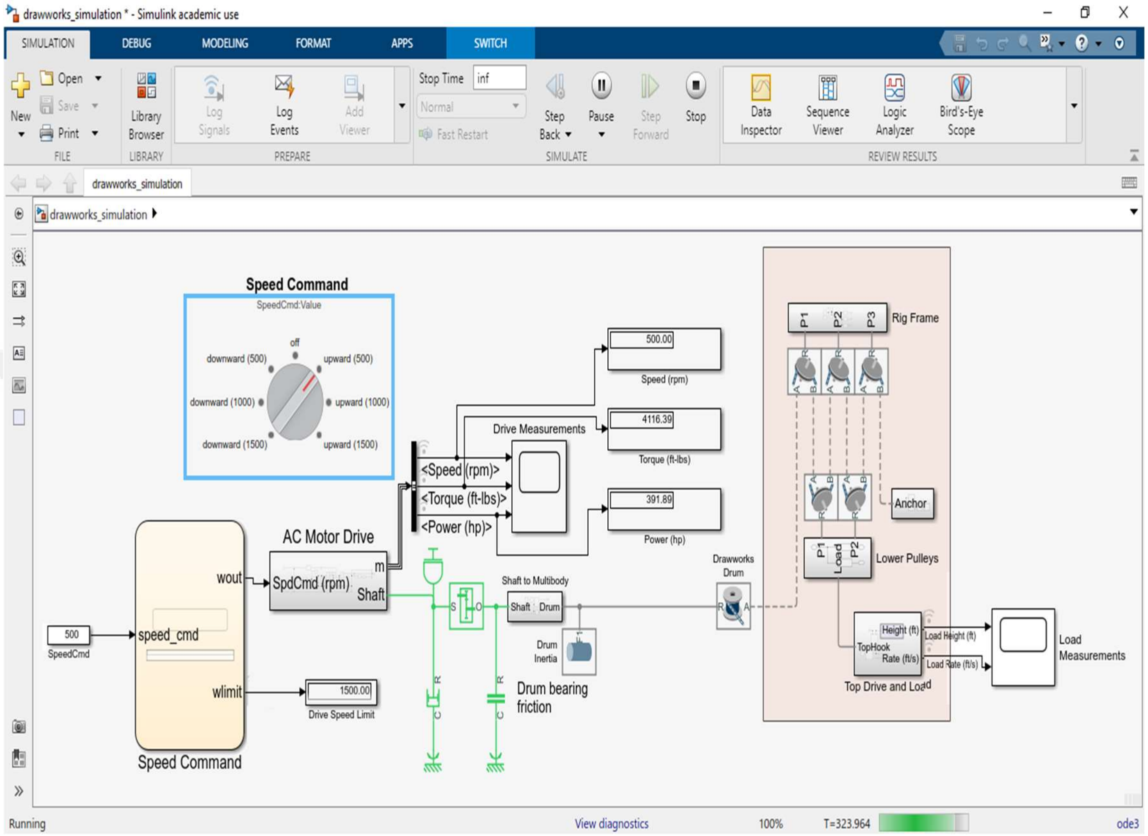


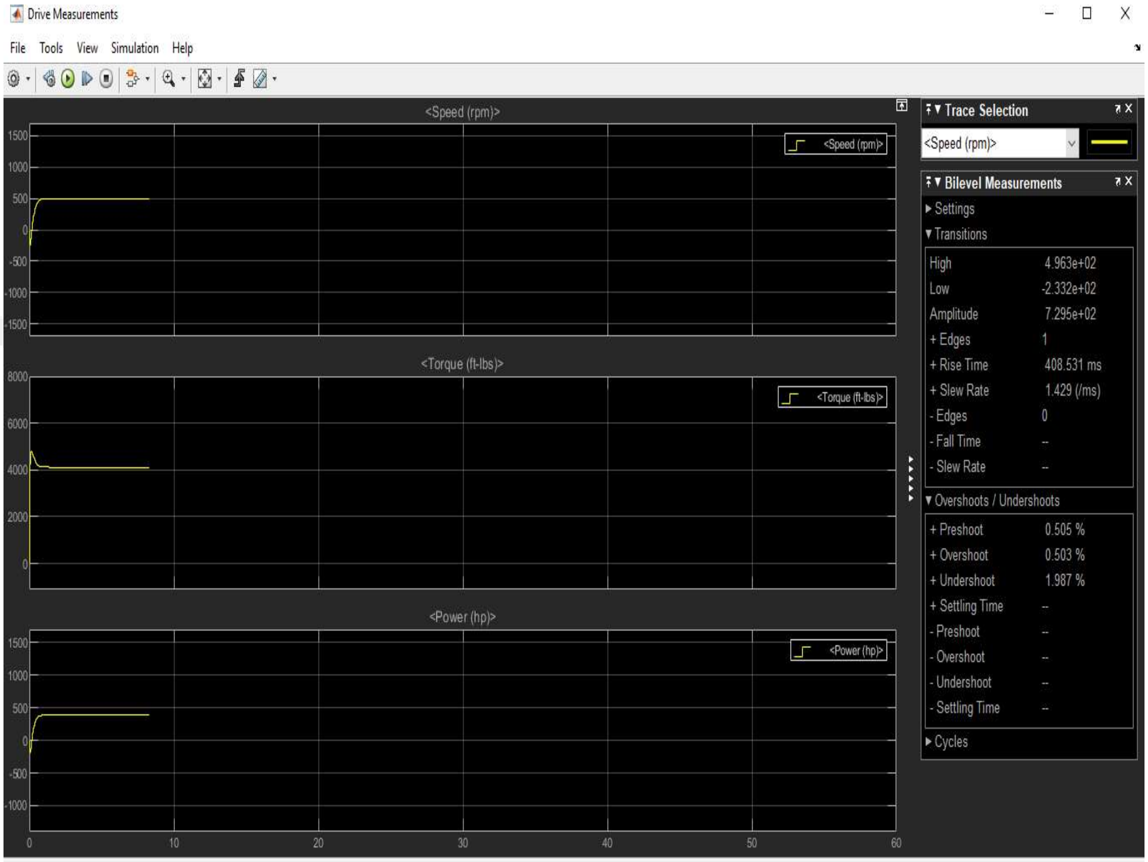
Figure 4.1 (e): Travelling block moving upward direction at 500 rpm

Travelling block is moving in upward direction. Speed has been set at 500 rpm. Torque is 4116.39 ft-lbs and Power is 391.89 hp.

Table 4.1 (e): Travelling block moving upward direction at 500 rpm

Speed (rpm)	500
Torque (ft-lbs)	4116.39
Power (hp)	391.89

I have attached below graph for speed, torque and power. In that graph rise time and overshoot for velocity has been measured.



<Speed (rpm)>

High	4.963e+02
Low	-2.332e+02
Amplitude	7.295e+02
+ Edges	1
+ Rise Time	408.531 ms
+ Slew Rate	1.429 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--
+ Preshoot	0.505 %
+ Overshoot	0.503 %
+ Undershoot	1.987 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Figure 4.1 (f): Graph for speed, torque, and power at 500 rpm

Table 4.1 (f): Rise time and overshoot at 500 rpm

Rise Time (ms)	408.531
Overshoot (%)	0.503

Rise Time for speed at 500 rpm is 408.531 ms and overshoot is 0.503% when travelling block is move upward direction.

b) Travelling block moving downward direction

Travelling block would be operated at different speed for observe the fall time and overshoot of velocity.

1) Travelling block moving downward direction at 1500 rpm

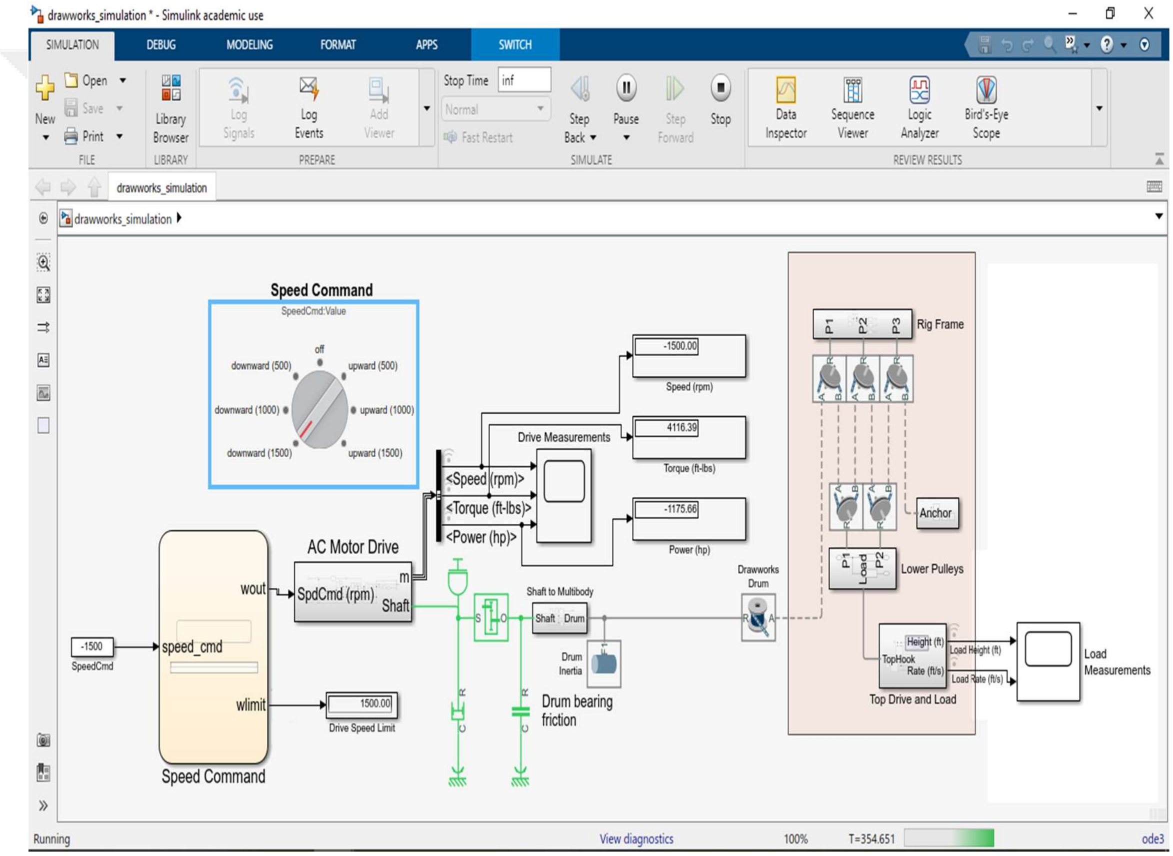


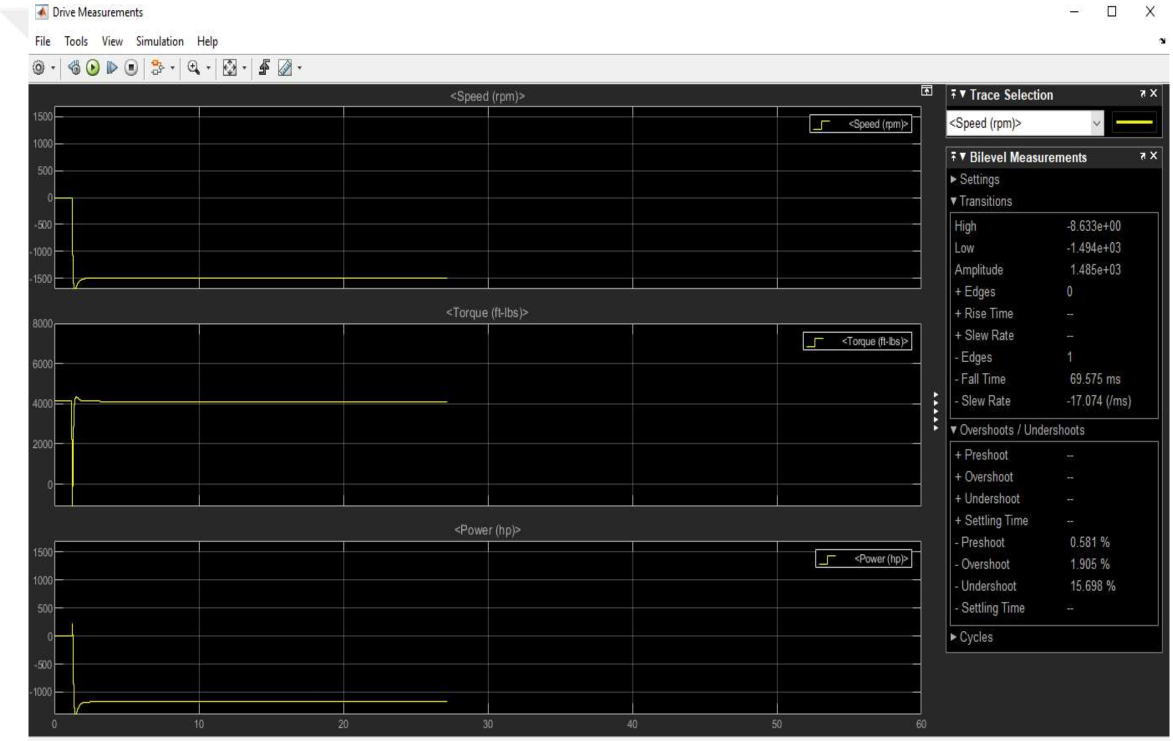
Figure 4.1 (g): Travelling block moving downward direction at 1500 rpm

Travelling block is moving in downward direction. Speed has been set at -1500 rpm, Torque is 4116.39 ft-lbs and Power is -1175.66 hp here negative sign shows the direction.

Table 4.1 (g): Travelling block moving downward direction at 1500 rpm

Speed (rpm)	1500
Torque (ft-lbs)	4116.39
Power (hp)	-1175.66

I have attached below graph for speed, torque and power. In that graph fall time and overshoot for velocity has been measured.



Transitions

High	-8.633e+00
Low	-1.494e+03
Amplitude	1.485e+03
+ Edges	0
+ Rise Time	--
+ Slew Rate	--
- Edges	1
- Fall Time	69.575 ms
- Slew Rate	-17.074 (/ms)

Overshoots / Undershoots

+ Preshoot	--
+ Overshoot	--
+ Undershoot	--
+ Settling Time	--
- Preshoot	0.581 %
- Overshoot	1.905 %
- Undershoot	15.698 %
- Settling Time	--

Figure 4.1 (h): Graph for speed, torque, and power at 1500 rpm

Table 4.1 (h): Fall time and overshoot at 1500 rpm

Fall Time (ms)	69.575
Overshoot (%)	1.905

Fall Time for speed at -1500 rpm is 69.575 ms and overshoot is 1.905% when travelling block is move downward direction.

2) Travelling block moving downward direction at 1000 rpm

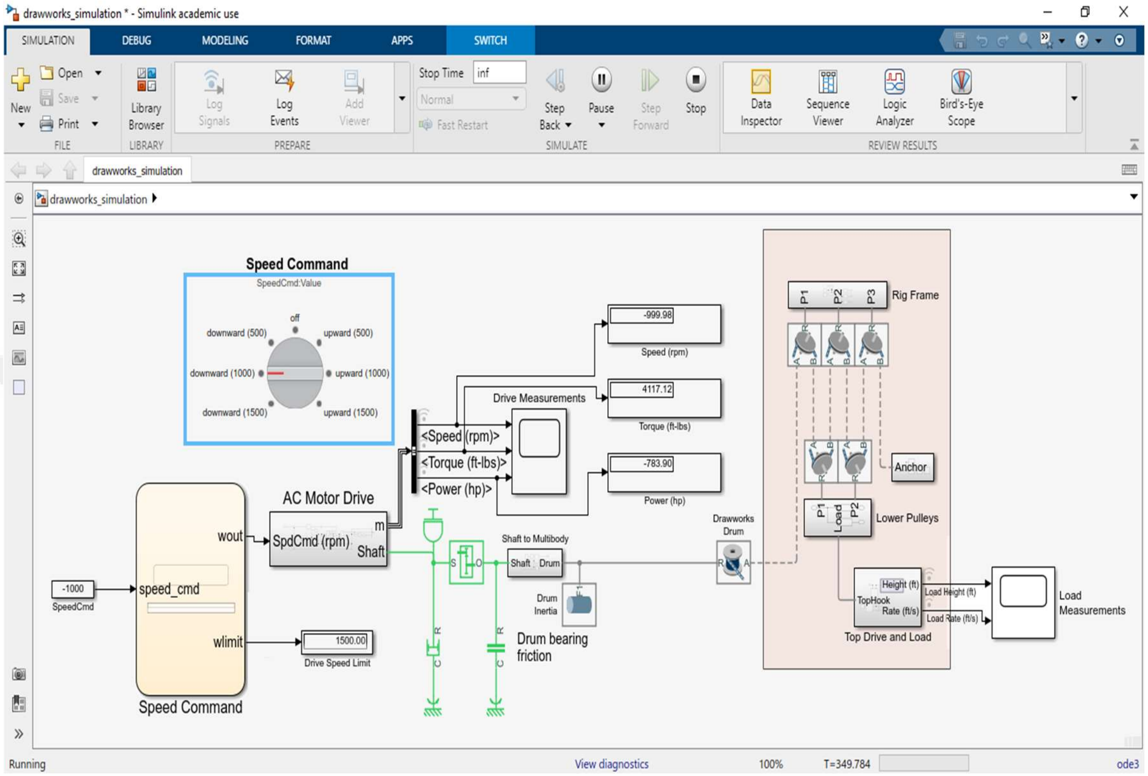


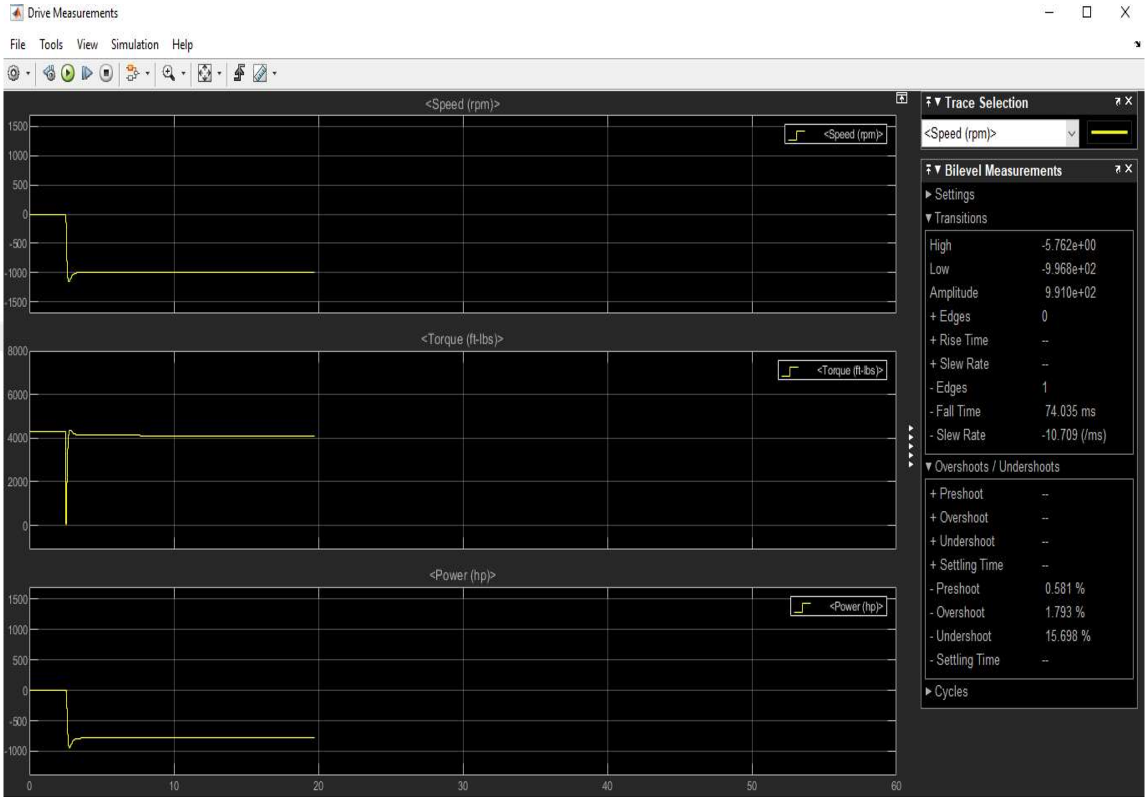
Figure 4.1 (i): Travelling block moving downward direction at 1000 rpm

Travelling block is moving in downward direction. Speed has been set at -1000 rpm, Torque is 4117.12 ft-lbs and Power is -783.90 hp here negative sign shows the direction.

Table 4.1 (i): Travelling block moving downward direction at 1000 rpm

Speed (rpm)	1000
Torque (ft-lbs)	4117.12
Power (hp)	-783.90

I have attached below graph for speed, torque and power. In that graph fall time and overshoot for velocity has been measured.



Transitions	
High	-5.762e+00
Low	-9.968e+02
Amplitude	9.910e+02
+ Edges	0
+ Rise Time	--
+ Slew Rate	--
- Edges	1
- Fall Time	74.035 ms
- Slew Rate	-10.709 (/ms)

Overshoots / Undershoots	
+ Preshoot	--
+ Overshoot	--
+ Undershoot	--
+ Settling Time	--
- Preshoot	0.581 %
- Overshoot	1.793 %
- Undershoot	15.698 %
- Settling Time	--

Figure 4.1 (j): Graph for speed, torque, and power at 1000 rpm

Table 4.1 (j): Fall time and overshoot at 1000 rpm

Fall Time (ms)	74.035
Overshoot (%)	1.793

Fall Time for speed at -1000 rpm is 74.035 ms and overshoot is 1.793% when travelling block is move downward direction.

3) Travelling block moving downward direction at 500 rpm

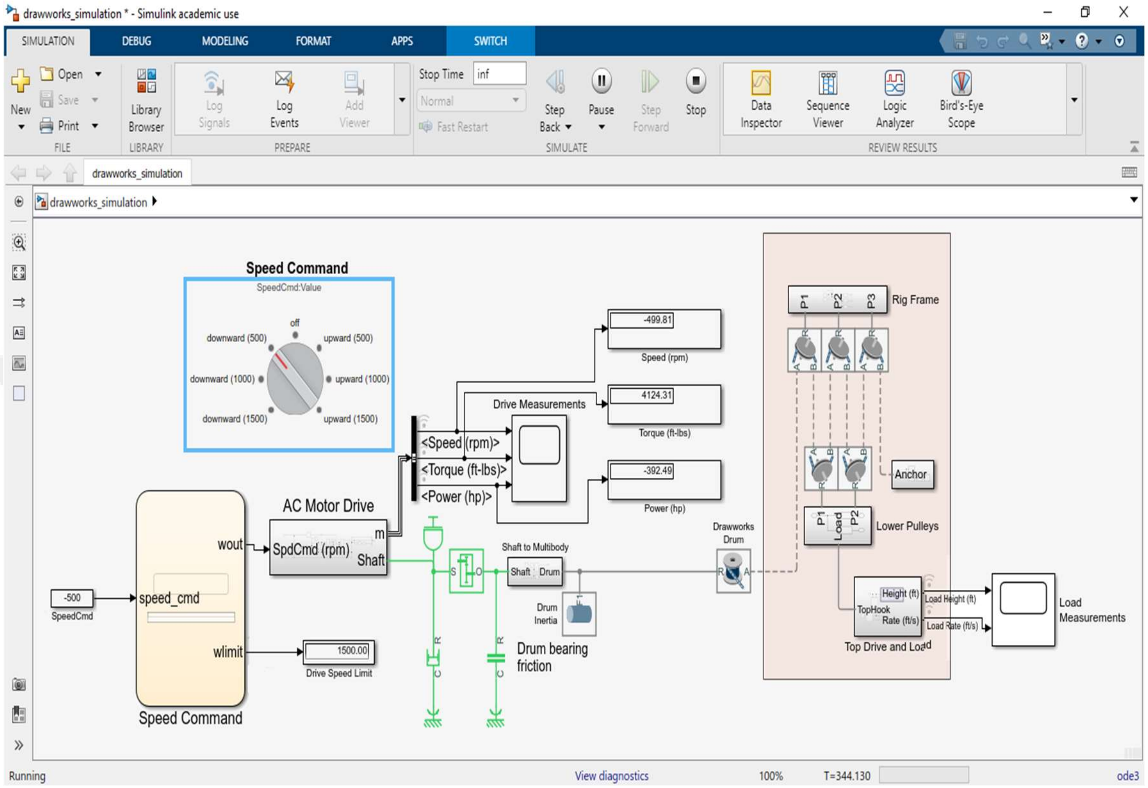


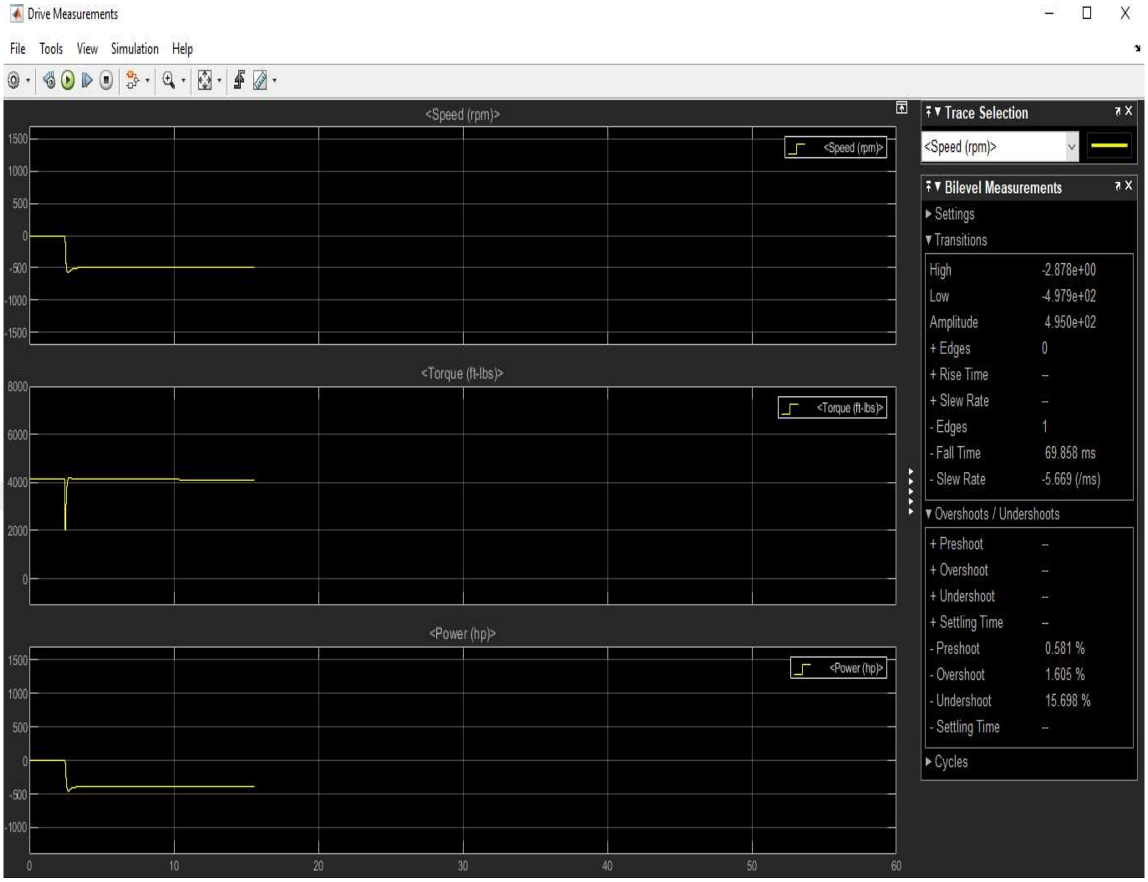
Figure 4.1 (k): Travelling block moving downward direction at 500 rpm

Travelling block is moving in downward direction. Speed has been set at -500 rpm, Torque is 4124.31 ft-lbs and Power is -392.49 hp here negative sign shows the direction.

Table 4.1 (k): Travelling block moving downward direction at 500 rpm

Speed (rpm)	500
Torque (ft-lbs)	4124.31
Power (hp)	-392.49

I have attached below graph for speed, torque and power. In that graph fall time and overshoot for velocity has been measured.



```
<Speed (rpm)>

Transitions
High              -2.878e+00
Low               -4.979e+02
Amplitude          4.950e+02
+ Edges            0
+ Rise Time        --
+ Slew Rate        --
- Edges            1
- Fall Time        69.858 ms
- Slew Rate        -5.669 (/ms)

Overshoots / Undershoots
+ Preshoot         --
+ Overshoot        --
+ Undershoot       --
+ Settling Time    --
- Preshoot         0.581 %
- Overshoot        1.605 %
- Undershoot       15.698 %
- Settling Time    --
```

Figure 4.1 (l): Graph for speed, torque, and power at 500 rpm

Table 4.1 (l): Fall time and overshoot at 500 rpm

Fall Time (ms)	69.858
Overshoot (%)	1.605

Fall Time for speed at -500 rpm is 69.858 ms and overshoot is 1.605% when travelling block is move downward direction.

4.2 Prototype model of Drawworks Control System

From previous chapters it has been observed that prototype model of drawworks is controlled by joystick. There are two directions for movement of travelling block that is either upward or downward. So, Joystick movement for control the drawworks is either in forward or in backward direction. Here both directions of joystick for control the drawworks and travelling block would be observed.

a) Joystick at Forward direction

Drawworks can only rotate either in forward direction or in backward direction. In this section forward direction of joystick signal would be observed. Deadman button is pressed and joystick is moved forward at maximum position. When joystick is in forward direction travelling block move in downward direction. I have attached below a diagram of drawworks at maximum forward direction of joystick.

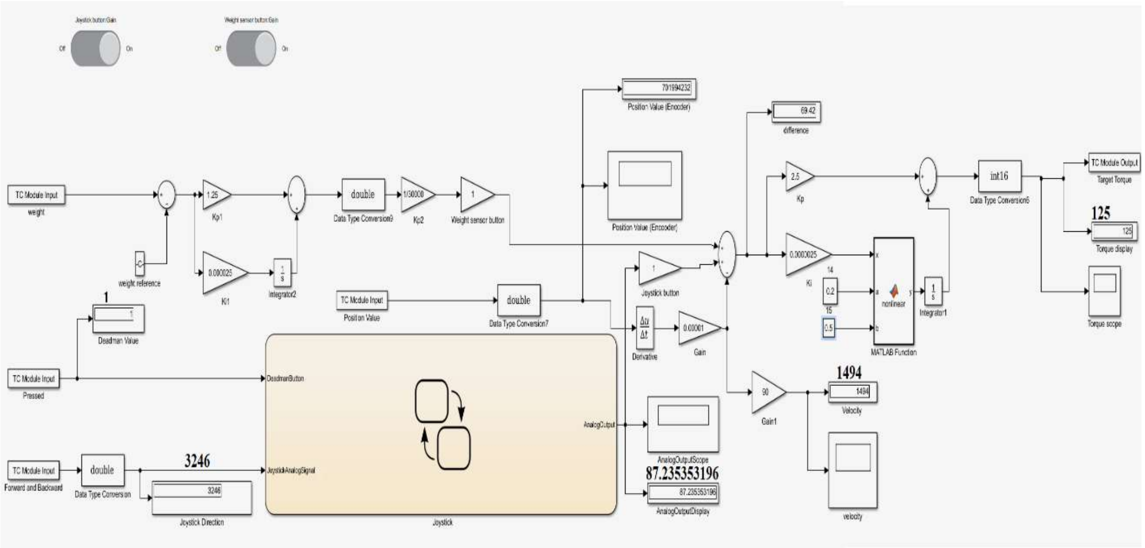


Figure 4.2 (a): Joystick at Forward direction

Above diagram shows the values when joystick is at maximum forward direction. It can be seen that deadman is pressed without it joystick would not give an analog output signal. At input port it can be seen that joystick is in forward direction and analog output signal of joystick is 87.235353196 here value is positive it shows the joystick forward direction, and the numerical value shows the joystick analog output signal. Position value is encoder value that is connected with derivative block, its output value is connected with joystick analog output signal and weight sensor by summation block**.** Summation block output is connected with PI controller to get steady state output. PI controller output is connected in series with convertor that convert signal into int16 as system required then it is connected with Tc Module output here required torque is achieved. I have attached below the table for above results.

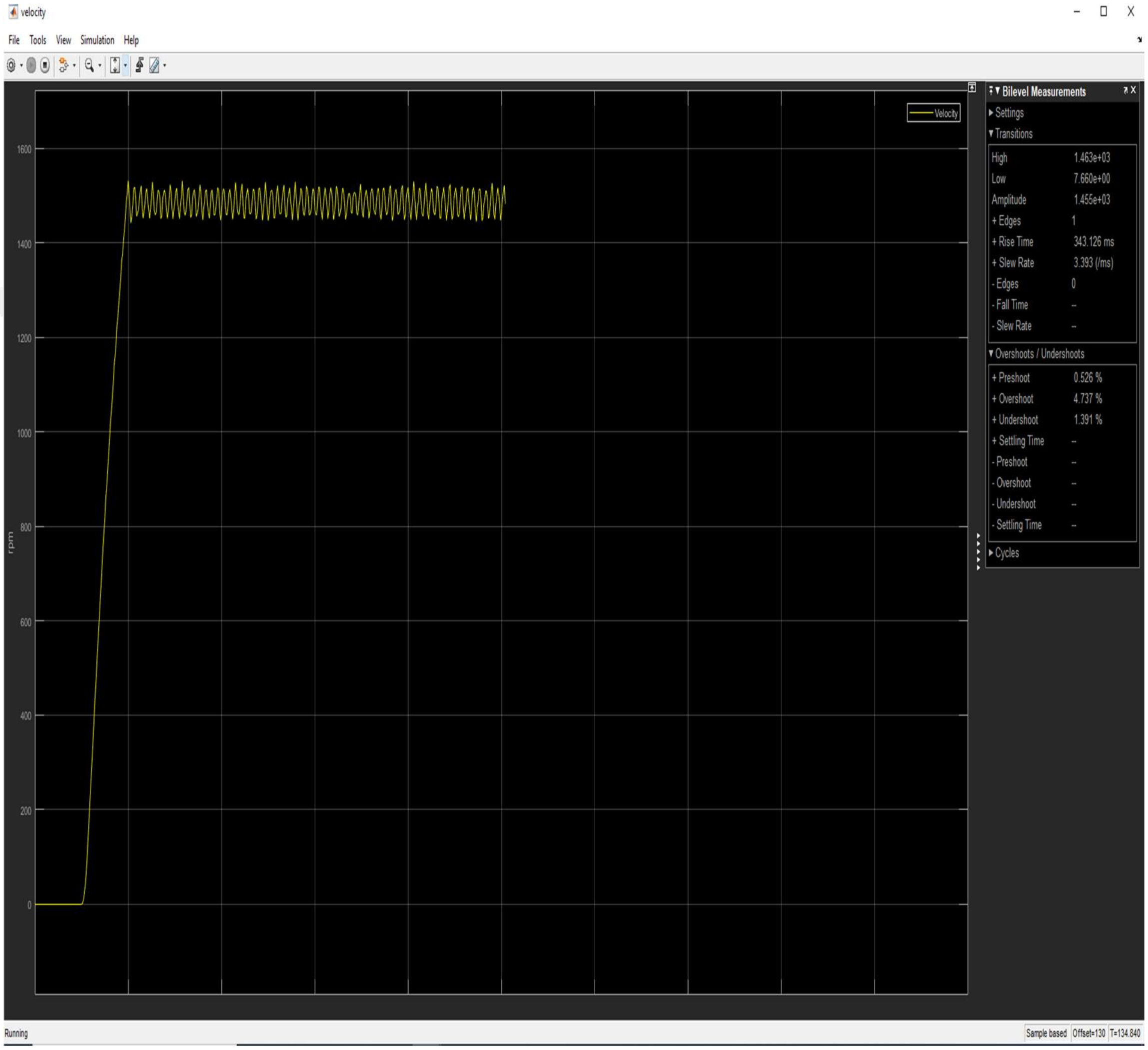
Table 4.2 (a): Joystick at Forward direction

Deadman Button	1
Joystick Analog input	3246
Analog Output	87.235353196
Velocity (rpm)	1494
Torque (N.m)	125

PI controller is used here to get steady state output at low rise time and minimum overshoot. For that PI controller is tuned using various Kp and Ki values for linear and non-linear PI controller to get steady state output. I have attached below the graph of velocity using linear and non-linear PI controller.

1) Kp=2.5, Ki=0.00000025

Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	1.463e+03
Low	7.660e+00
Amplitude	1.455e+03
+ Edges	1
+ Rise Time	343.126 ms
+ Slew Rate	3.393 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--

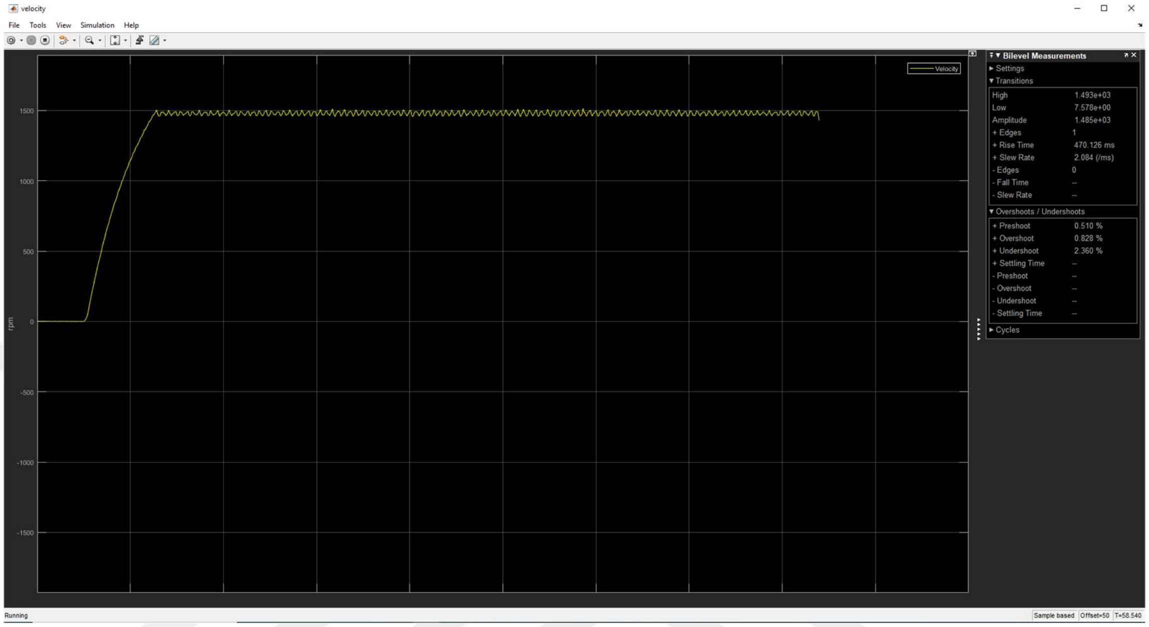
Overshoots / Undershoots

+ Preshoot	0.526 %
+ Overshoot	4.737 %
+ Undershoot	1.391 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Cycles

Figure 4.2 (b): Velocity using Linear PI Controller tune at Kp=2.5, Ki=0.00000025.

Non-Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	1.493e+03
Low	7.578e+00
Amplitude	1.485e+03
+ Edges	1
+ Rise Time	470.126 ms
+ Slew Rate	2.084 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--

Overshoots / Undershoots

+ Preshoot	0.510 %
+ Overshoot	0.828 %
+ Undershoot	2.360 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Cycles

Figure 4.2(c): Velocity using Non-Linear PI Controller tune at Kp=2.5, Ki=0.00000025

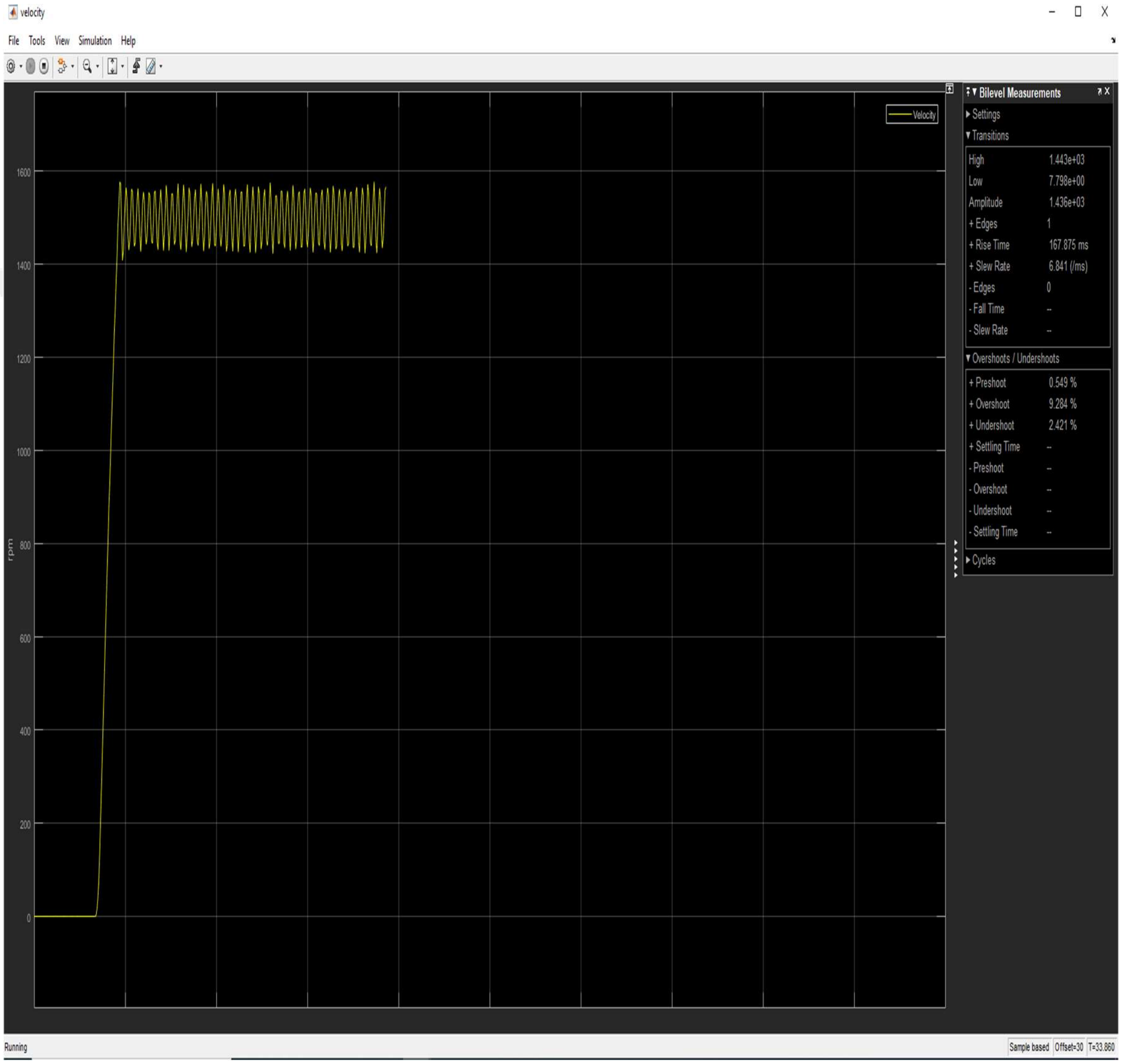
Table 4.2 (b): Rise Time and overshoot using linear and Non- Linear PI Controller tune at Kp=2.5, Ki=0.00000025

Linear PI Controller		Non-Linear PI Controller	
Rise Time (ms)	343.126	**Rise Time (ms)**	470.126
Overshoot (%)	4.737	**Overshoot (%)**	0.828

Above graph is of motor velocity. PI controller has been tuned at Kp=2.5, Ki=0.00000025 for linear and non-linear to get steady state. It can be observed that rise time for linear PI controller velocity is less than the rise time for non-linear PI controller velocity, and overshoot for linear PI controller velocity is greater than the overshoot for non-linear PI controller velocity. So, for rise time Linear PI controller is good and for minimum overshoot non-linear PI controller is good.

2) Kp=4, Ki=0.000005

Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	1.443e+03
Low	7.798e+00
Amplitude	1.436e+03
+ Edges	1
+ Rise Time	167.875 ms
+ Slew Rate	6.841 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--

Overshoots / Undershoots

+ Preshoot	0.549 %
+ Overshoot	9.284 %
+ Undershoot	2.421 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Cycles

Sample based | Offset=30 | T=33.860

Figure 4.2(d): Velocity using Linear PI Controller tune at Kp=4, Ki=0.000005.

Non-Linear PI Controller

Bilevel Measurements

Settings

Transitions

High	1.521e+03
Low	7.630e+00
Amplitude	1.513e+03
+ Edges	1
+ Rise Time	263.180 ms
+ Slew Rate	4.599 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--

Overshoots / Undershoots

+ Preshoot	0.510 %
+ Overshoot	1.531 %
+ Undershoot	5.637 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Cycles

Figure 4.2(e): Velocity using Non-Linear PI Controller tune at Kp=4, Ki=0.000005

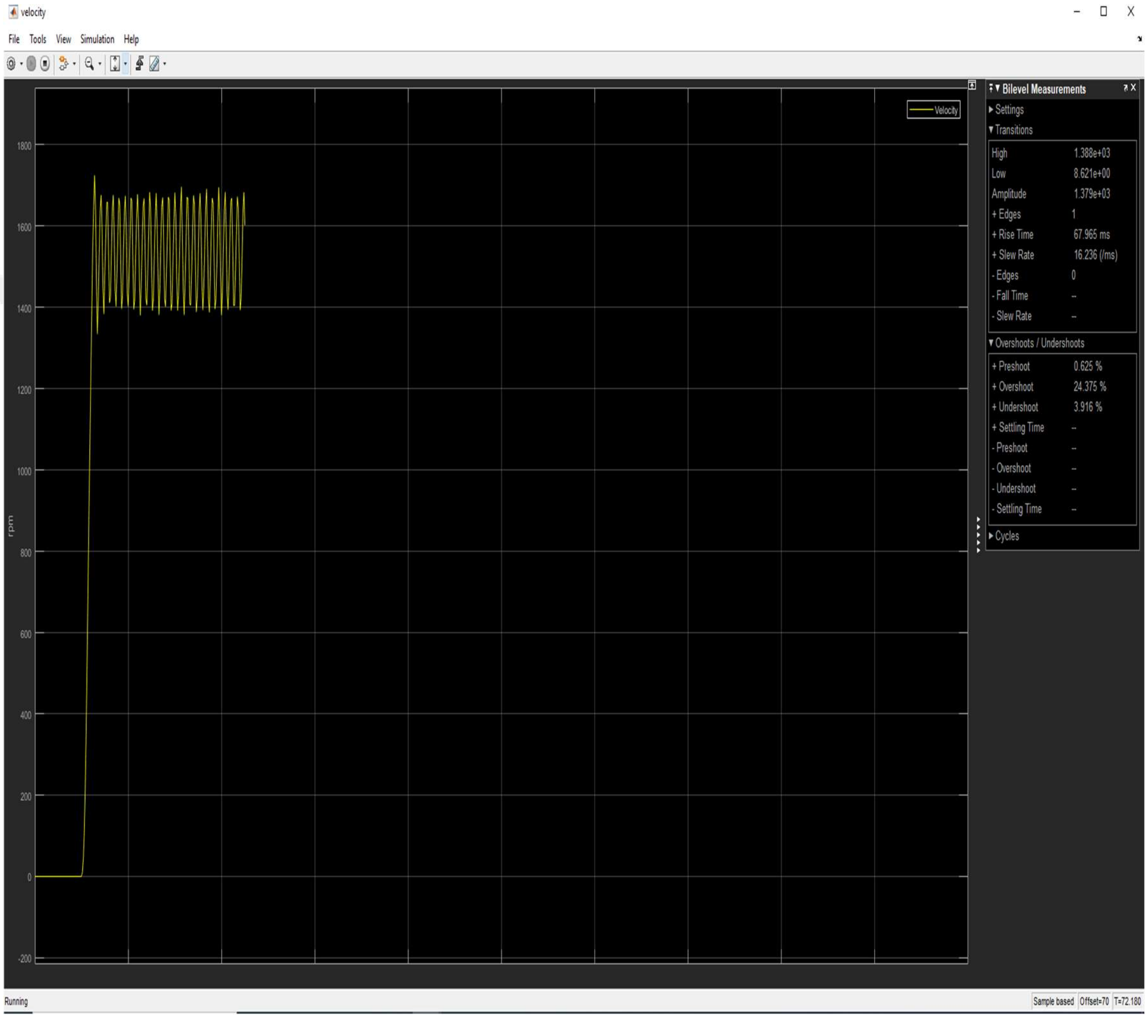
Table 4.2 (c): Rise Time and overshoot using linear and Non- Linear PI Controller tune at Kp=4, Ki=0.000005

Linear PI Controller		**Non-Linear PI Controller**	
Rise Time (ms)	167.875	**Rise Time (ms)**	263.180
Overshoot (%)	9.284	**Overshoot (%)**	1.531

Above graph is of motor velocity. PI controller has been tuned at Kp=4, Ki=0.000005 for linear and non-linear to get steady state. It can be observed that rise time for linear PI controller velocity is less than the rise time for non-linear PI controller velocity, and overshoot for linear PI controller velocity is greater than the overshoot for non-linear PI controller velocity. So, rise time of Linear PI controller is good and for minimum overshoot non-linear PI controller is good.

3) Kp=8.5, Ki=0.00000025

Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	1.388e+03
Low	8.621e+00
Amplitude	1.379e+03
+ Edges	1
+ Rise Time	67.965 ms
+ Slew Rate	16.236 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--

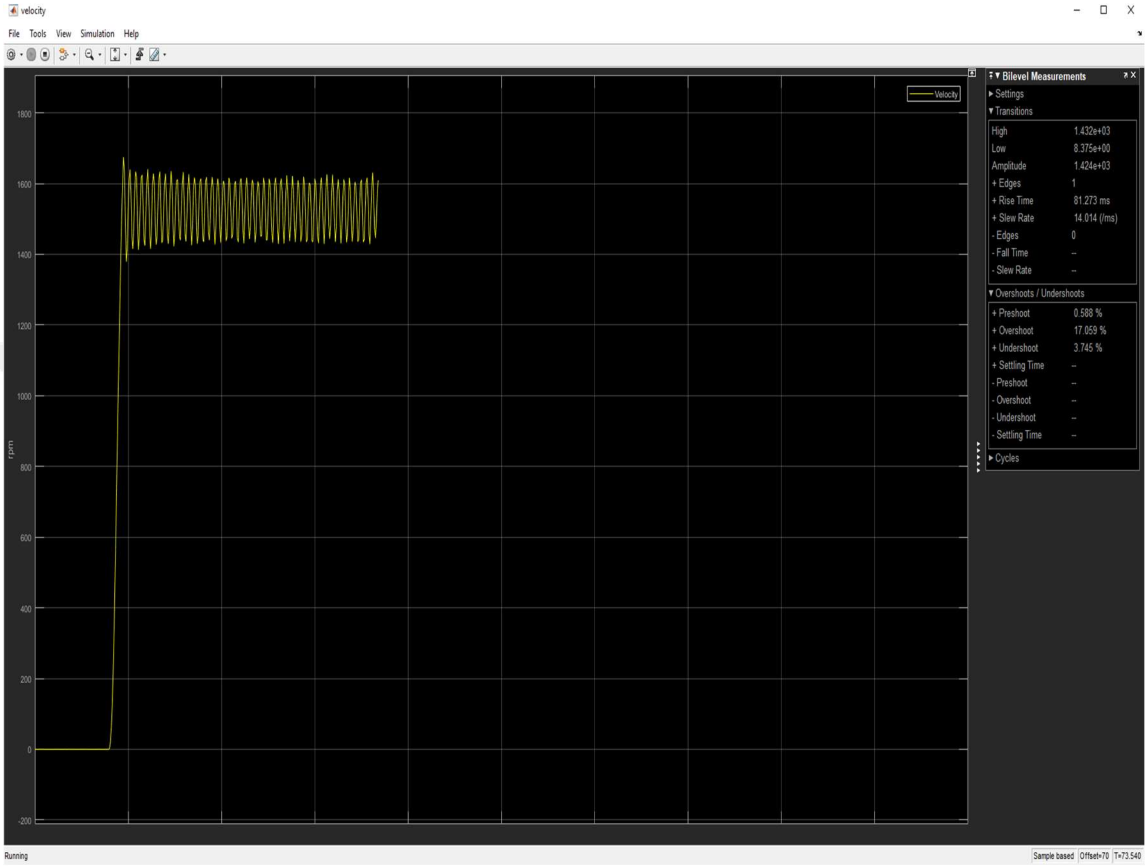
Overshoots / Undershoots

+ Preshoot	0.625 %
+ Overshoot	24.375 %
+ Undershoot	3.916 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Cycles

Figure 4.2(f): Velocity using Linear PI Controller tune at Kp=8.5, Ki=0.00000025.

Non-Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	1.432e+03
Low	8.375e+00
Amplitude	1.424e+03
+ Edges	1
+ Rise Time	81.273 ms
+ Slew Rate	14.014 (/ms)
- Edges	0
- Fall Time	--
- Slew Rate	--

Overshoots / Undershoots

+ Preshoot	0.588 %
+ Overshoot	17.059 %
+ Undershoot	3.745 %
+ Settling Time	--
- Preshoot	--
- Overshoot	--
- Undershoot	--
- Settling Time	--

Cycles

Figure 4.2(g): Velocity using Non- Linear PI Controller tune at Kp=8.5, Ki=0.00000025

Table 4.2 (d): Rise Time and overshoot using linear and Non- Linear PI Controller tune at Kp=8.5, Ki=0.00000025

Linear PI Controller		**Non-Linear PI Controller**	
Rise Time (ms)	67.965	**Rise Time (ms)**	81.273
Overshoot (%)	24.375	**Overshoot (%)**	17.059

Above graph is of motor velocity. PI controller has been tuned at Kp=8.5, Ki=0.00000025; for linear and non-linear to get steady state. It can be observed that rise time for linear PI controller is less than the rise time for non-linear PI controller velocity, and overshoot for linear PI controller velocity is greater than the overshoot for non-linear PI controller velocity. So, rise time of Linear PI controller is good and for minimum overshoot non-linear PI controller is good.

b) Joystick at Backward direction

When joystick is at backward direction travelling block moves upward. I have attached below the graph of travelling block moving upward direction. I have tuned the PI controller by varying Kp and Ki values to get desired output.

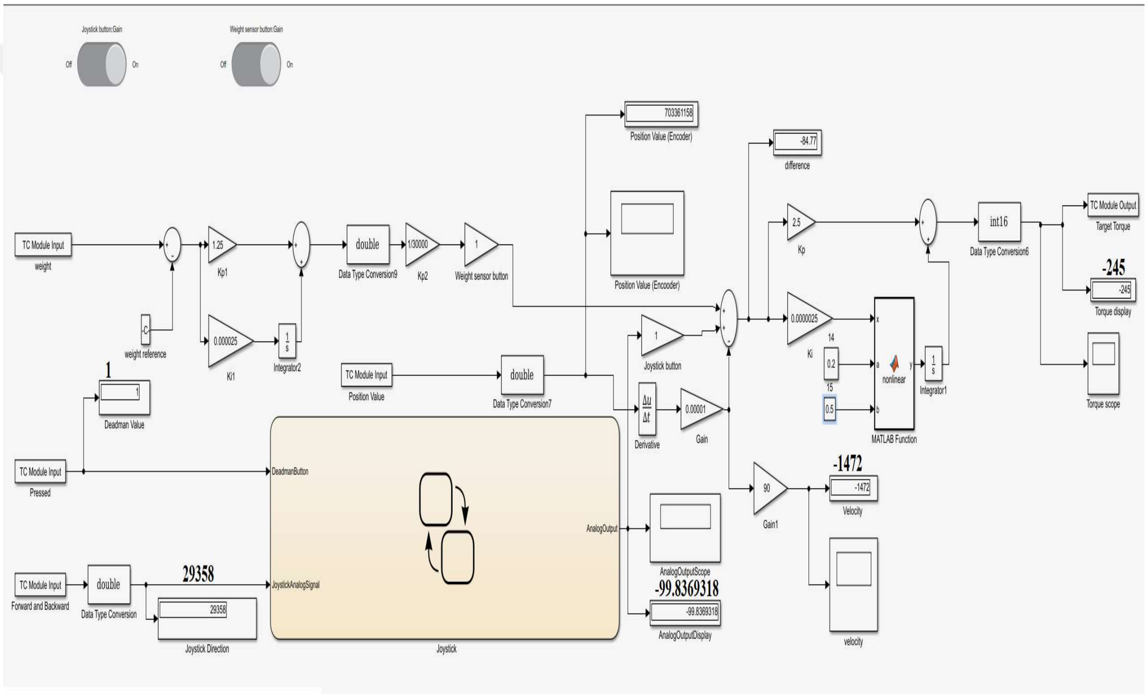


Figure 4.2 (h): Joystick at Backward Direction

Above diagram shows the values when joystick is at maximum backward direction. It can be seen that deadman is pressed without it joystick would not give an analog signal output. From input port it can be seen that joystick is in backward direction and analog output signal of joystick is -99.8369318 here negative sign shows the backward direction, and the numerical value shows the joystick analog output signal. Position value is encoder value that is connected with derivative block its output value is connected with joystick analog output signal and weight sensor signal by summation block**.** Output of summation block is connected with PI controller to get steady state output. Output of PI controller is connected in series with convertor that converts signal into int16 as system required this form of signal after conversion it is connected with Tc Module output here desired torque is achieved. I have attached below the table for results which can be observed from above diagram.

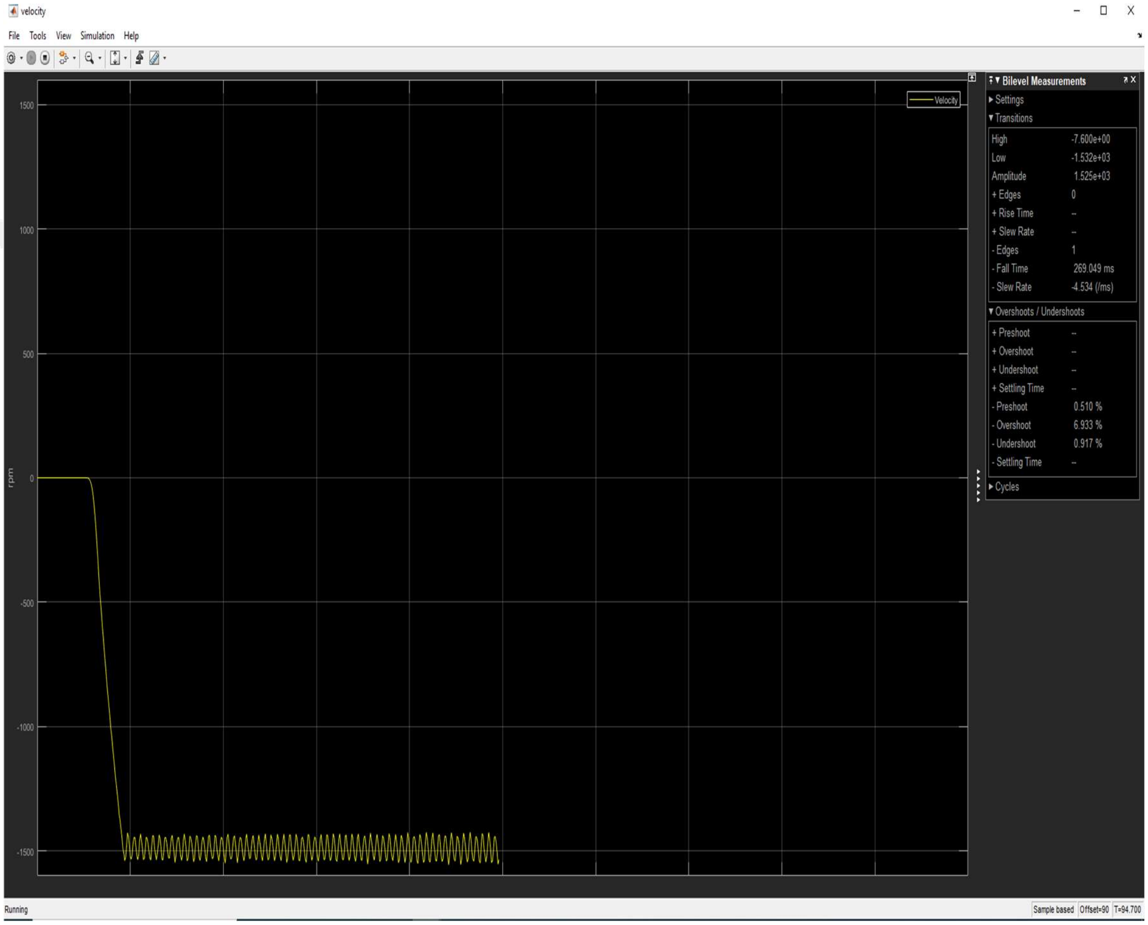
Table 4.2 (e): Joystick at Backward Direction

Deadman Button	1
Joystick Analog input	29358
Analog Output	-99.8369318
Velocity	-1472
Torque	-245

Various Kp and Ki values has been tried for tune the linear and non-linear PI controller to get steady state output, minimum fall time and overshoot. I have attached below the graph of velocity using linear and non-linear PI controller.

1) Kp=2.5, Ki=0.00000025

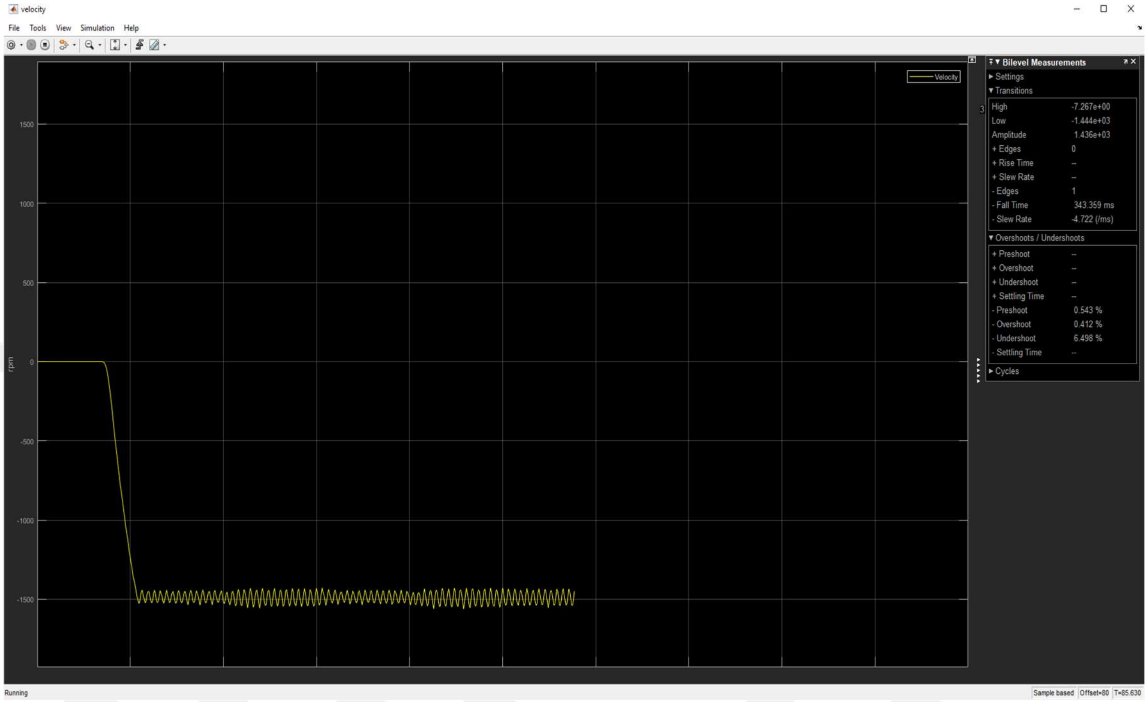
Linear PI Controller



```
Bilevel Measurements
▶ Settings
▼ Transitions
High             -7.600e+00
Low              -1.532e+03
Amplitude         1.525e+03
+ Edges          0
+ Rise Time      --
+ Slew Rate      --
- Edges          1
- Fall Time       269.049 ms
- Slew Rate      -4.534 (/ms)
▼ Overshoots / Undershoots
+ Preshoot       --
+ Overshoot      --
+ Undershoot     --
+ Settling Time  --
- Preshoot        0.510 %
- Overshoot       6.933 %
- Undershoot      0.917 %
- Settling Time  --
▶ Cycles
```

Figure 4.2 (i): Velocity using Linear PI Controller tune at Kp=2.5, Ki=0.00000025

Non-Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	-7.267e+00
Low	-1.444e+03
Amplitude	1.436e+03
+ Edges	0
+ Rise Time	--
+ Slew Rate	--
- Edges	1
- Fall Time	343.359 ms
- Slew Rate	-4.722 (/ms)

Overshoots / Undershoots

+ Preshoot	--
+ Overshoot	--
+ Undershoot	--
+ Settling Time	--
- Preshoot	0.543 %
- Overshoot	0.412 %
- Undershoot	6.498 %
- Settling Time	--

Cycles

Figure 4.2 (j): Velocity using Non-Linear PI Controller tune at Kp=2.5, Ki=0.00000025

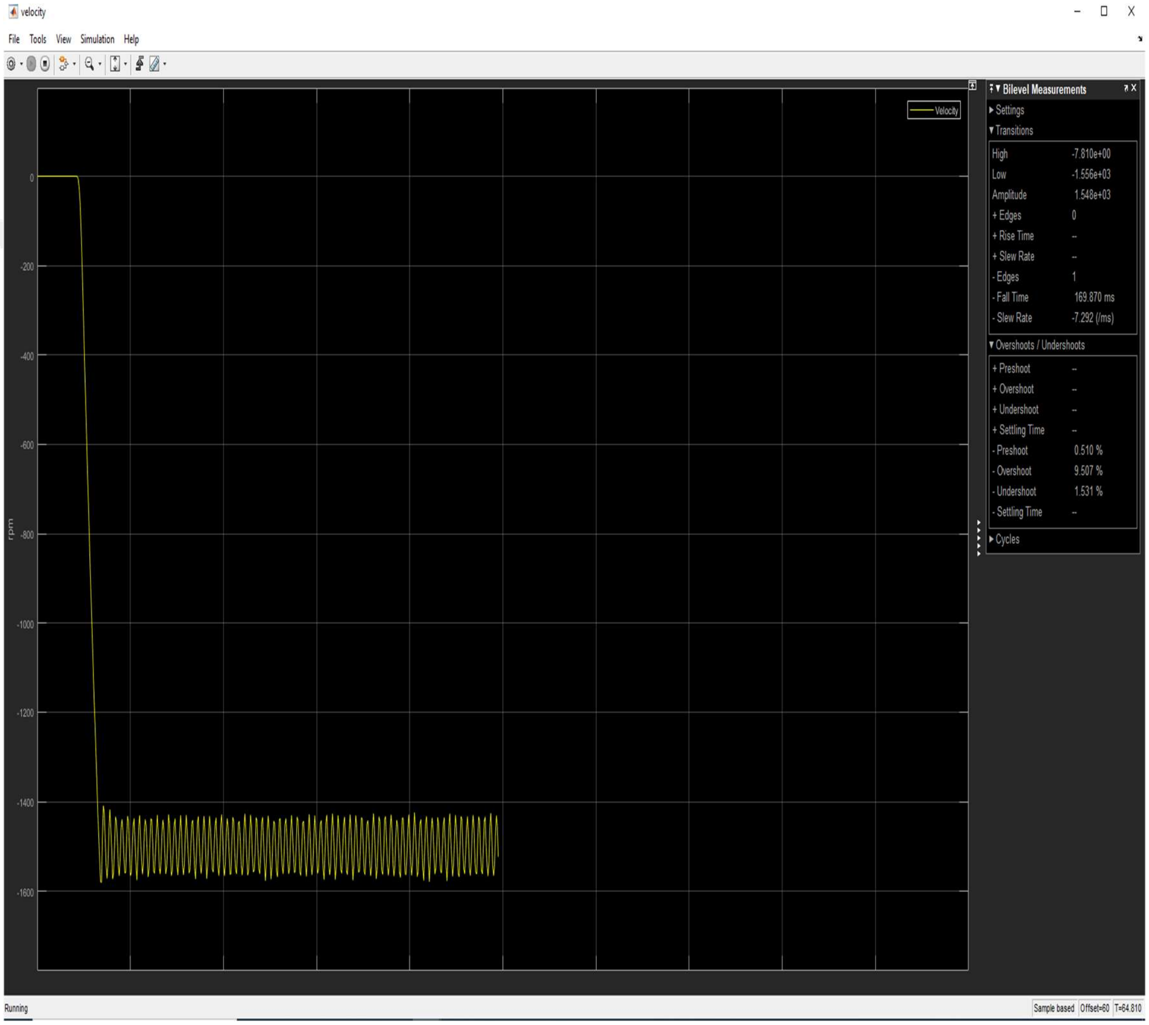
Table 4.2 (f): Fall Time and overshoot using linear and Non- Linear PI Controller tune at Kp=2.5, Ki=0.00000025

Linear PI Controller		**Non-Linear PI Controller**	
Fall Time (ms)	269.049	**Rise Time (ms)**	343.359
Overshoot (%)	6.933	**Overshoot (%)**	0.412

Above graph is of motor velocity. PI controller has been tuned at Kp=2.5, Ki=0.00000025 for linear and non-linear to get steady state. It can be observed that fall time for linear PI controller is less than the fall time for non-linear PI controller velocity, and overshoot for linear PI controller velocity is greater than the overshoot for non-linear PI controller velocity. So, fall time of Linear PI controller is good and for minimum overshoot non-linear PI controller is good.

2) Kp=4, Ki=0.000005

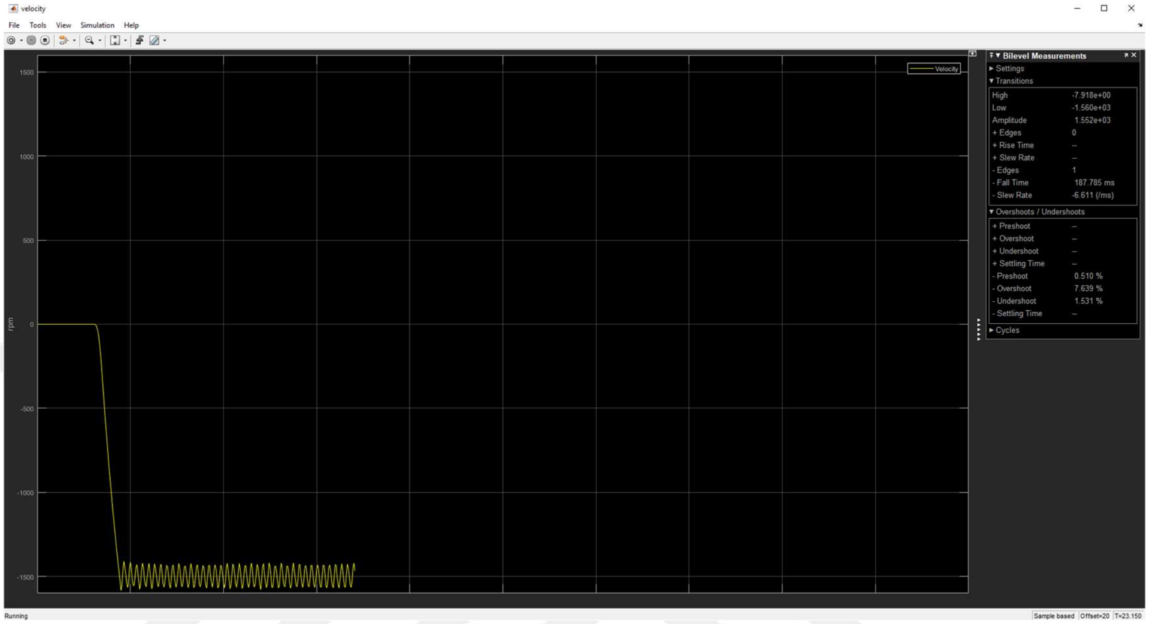
Linear PI Controller



```
Bilevel Measurements
▶ Settings
▼ Transitions
High            -7.810e+00
Low             -1.556e+03
Amplitude        1.548e+03
+ Edges          0
+ Rise Time      --
+ Slew Rate      --
- Edges          1
- Fall Time      169.870 ms
- Slew Rate     -7.292 (/ms)
▼ Overshoots / Undershoots
+ Preshoot       --
+ Overshoot      --
+ Undershoot     --
+ Settling Time  --
- Preshoot       0.510 %
- Overshoot      9.507 %
- Undershoot     1.531 %
- Settling Time  --
▶ Cycles
```

Figure 4.2(k): Velocity using Linear PI Controller tune at Kp=4, Ki=0.000005

Non-Linear PI Controller



```
Bilevel Measurements
► Settings
▼ Transitions
High              -7.918e+00
Low               -1.560e+03
Amplitude          1.552e+03
+ Edges           0
+ Rise Time       --
+ Slew Rate       --
- Edges           1
- Fall Time        187.785 ms
- Slew Rate       -6.611 (/ms)
▼ Overshoots / Undershoots
+ Preshoot        --
+ Overshoot       --
+ Undershoot      --
+ Settling Time   --
- Preshoot         0.510 %
- Overshoot        7.639 %
- Undershoot       1.531 %
- Settling Time   --
► Cycles
```

Figure 4.2(l): Velocity using Non-Linear PI Controller tune at Kp=4, Ki=0.000005

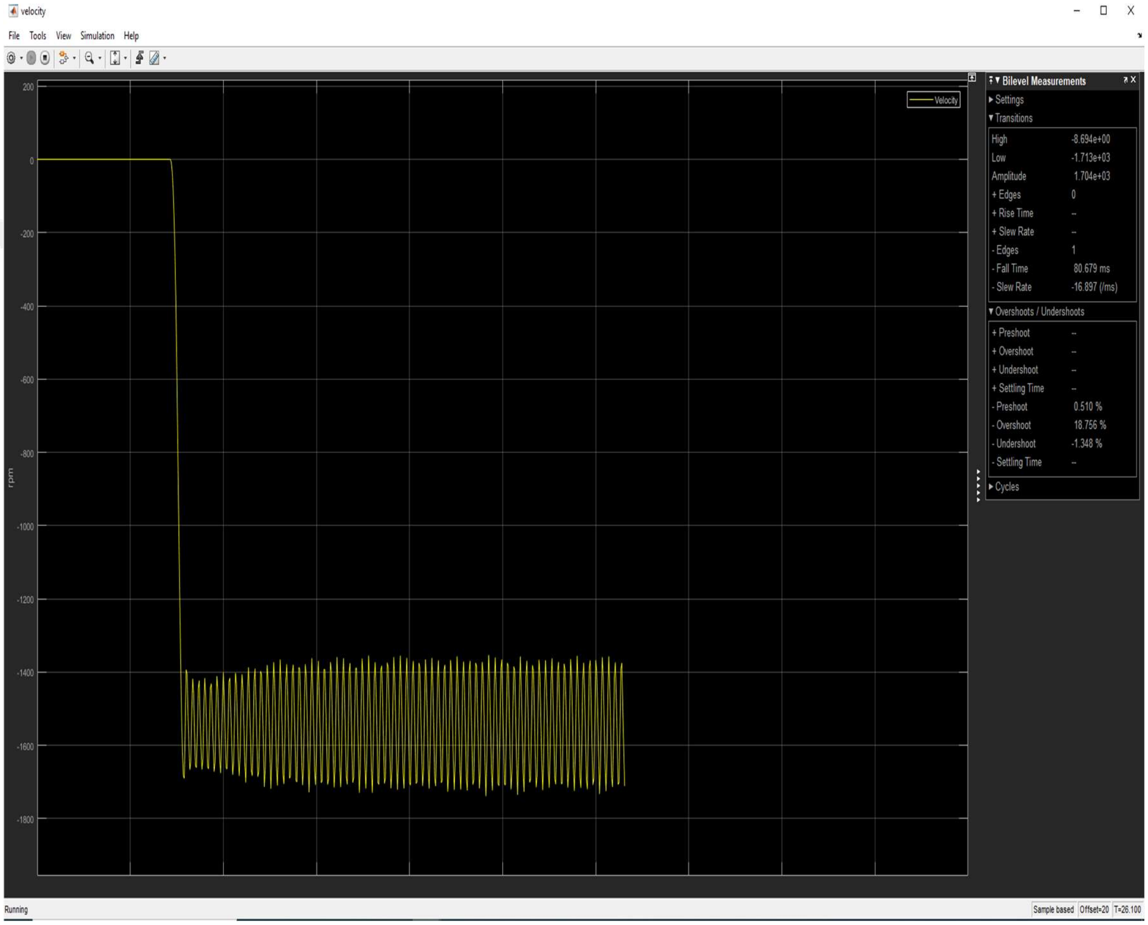
Table 4.2 (g): Fall Time and overshoot using linear Non- Linear PI Controller tune at Kp=4, Ki=0.000005

Linear PI Controller		**Non-Linear PI Controller**	
Fall Time (ms)	169.870	**Fall Time (ms)**	187.785
Overshoot (%)	9.507	**Overshoot (%)**	7.639

Above graph is of motor velocity. PI controller has been tuned at Kp=4; Ki=0.000005 for linear and non-linear to get steady state. It can be observed that fall time for linear PI controller velocity is lesser than the fall time for non-linear PI controller velocity, and overshoot for linear PI controller velocity is greater than the overshoot for non-linear PI controller velocity. So, for fall time Linear PI and for minimum overshoot at above tune non-linear PI controller is good.

3) Kp=8.5, Ki=0.00000025

Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	-8.694e+00
Low	-1.713e+03
Amplitude	1.704e+03
+ Edges	0
+ Rise Time	--
+ Slew Rate	--
- Edges	1
- Fall Time	80.679 ms
- Slew Rate	-16.897 (/ms)

Overshoots / Undershoots

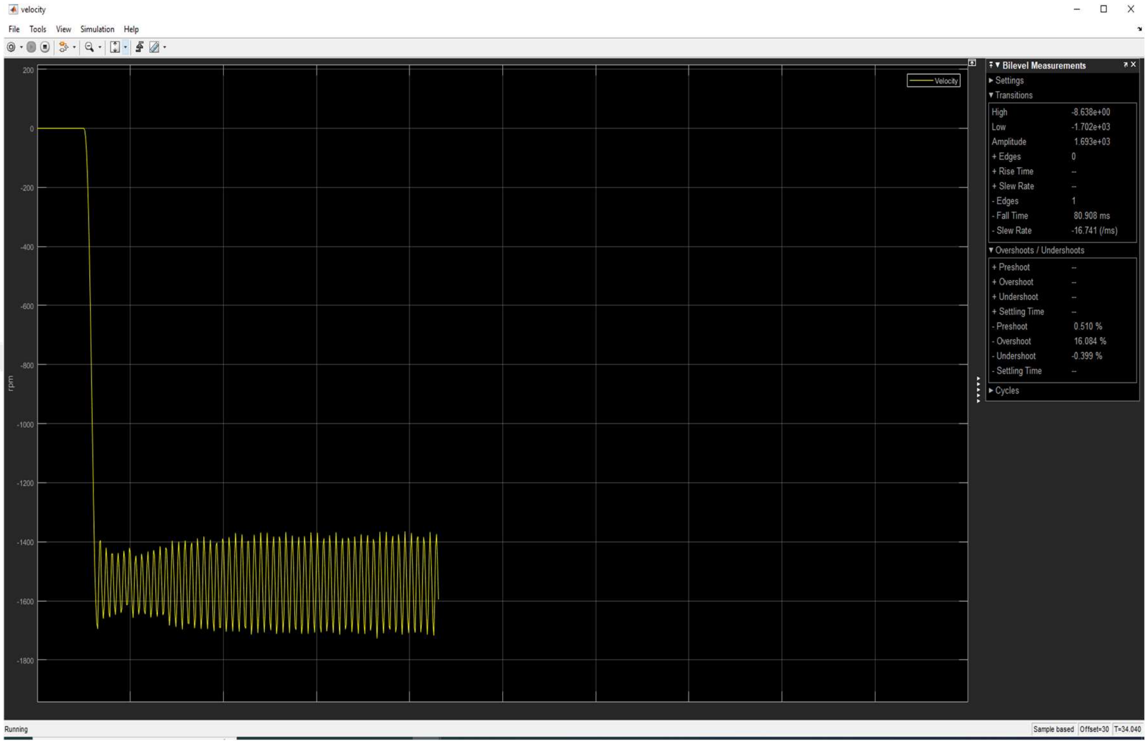
+ Preshoot	--
+ Overshoot	--
+ Undershoot	--
+ Settling Time	--
- Preshoot	0.510 %
- Overshoot	18.756 %
- Undershoot	-1.348 %
- Settling Time	--

Cycles

Sample based Offset=20 T=26.100

Figure 4.2(m): Velocity using Linear PI Controller tune at Kp=8.5, Ki=0.00000025

Non-Linear PI Controller



Bilevel Measurements

Settings

Transitions

High	-8.638e+00
Low	-1.702e+03
Amplitude	1.693e+03
+ Edges	0
+ Rise Time	--
+ Slew Rate	--
- Edges	1
- Fall Time	80.908 ms
- Slew Rate	-16.741 (/ms)

Overshoots / Undershoots

+ Preshoot	--
+ Overshoot	--
+ Undershoot	--
+ Settling Time	--
- Preshoot	0.510 %
- Overshoot	16.084 %
- Undershoot	-0.399 %
- Settling Time	--

Cycles

Figure 4.2(n): Velocity using Non-Linear PI Controller tune at Kp=8.5, Ki=0.00000025

Table 4.2 (h): Fall Time and overshoot using linear Non- Linear PI Controller tune at Kp=8.5, Ki=0.00000025

Linear PI Controller		**Non-Linear PI Controller**	
Fall Time (ms)	80.679	**Fall Time (ms)**	80.908
Overshoot (%)	18.756	**Overshoot (%)**	16.084

Above graph is of motor velocity. PI controller has been tuned at Kp=8.5, Ki=0.00000025 for linear and non-linear to get steady state. It can be observed that fall time for linear PI controller velocity is greater than the fall time for non-linear PI controller velocity, and overshoot for nonlinear PI controller velocity is less than the overshoot for linear PI controller velocity. So, fall time of linear PI controller is good and for minimum overshoot non-linear PI controller is good for above tuning.

I have attached below the combined table of results for rise & fall time and overshoot of all above results.

Table 4.2 (i): Joystick at Forward direction

Linear PI Controller		**Non-Linear PI Controller**	
Kp=2.5, Ki=0.00000025		**Kp=2.5, Ki=0.00000025**	
Rise Time (ms)	343.126	**Rise Time (ms)**	470.126
Overshoot (%)	4.737	**Overshoot (%)**	0.828
Kp=4, Ki=0.000005		**Kp=4, Ki=0.000005**	
Rise Time (ms)	167.875	**Rise Time (ms)**	263.180
Overshoot (%)	9.284	**Overshoot (%)**	1.531
Kp=8.5, Ki=0.00000025		**Kp=8.5, Ki=0.00000025**	
Rise Time (ms)	67.965	**Rise Time (ms)**	81.273
Overshoot (%)	24.375	**Overshoot (%)**	17.059

Table 4.2 (j): Joystick at Backward direction

Linear PI Controller		**Non-Linear PI Controller**	
Kp=2.5, Ki=0.00000025		**Kp=2.5, Ki=0.00000025**	
Fall Time (ms)	269.049	**Fall Time (ms)**	343.359
Overshoot (%)	6.933	**Overshoot (%)**	0.412
Kp=4, Ki=0.000005		**Kp=4, Ki=0.000005**	
Fall Time (ms)	169.870	**Fall Time (ms)**	187.785

Overshoot (%)	9.507	**Overshoot (%)**	7.639
Kp=8.5, Ki=0.00000025		**Kp=8.5, Ki=0.00000025**	
Fall Time (ms)	80.679	**Fall Time (ms)**	80.908
Overshoot (%)	18.756	**Overshoot (%)**	16.084

From above tables it can be observed that at Kp=2.5, Ki=0.00000025 non-linear PI Controller have less overshoot and desired rise & fall time when it is compared with other results. So desired results are achieved at Kp=2.5, Ki=0.00000025 by non-linear PI Controller.

CHAPTER 5

CONCLUSION

Simulation based model and prototype model for control system of drawworks has been observed. In simulation based model results and graphs for velocity, torque and power were observed. Rise time and overshoot for velocity were calculated on different speed value when travelling block was moving upward. Also, fall time and overshoot for velocity was calculated on different speed value when travelling block was moving downward. When travelling block reached its maximum or minimum height that is 50 ft and -250 ft driller has to change the direction of load movement. If direction is not changed system would try to move further for example if travelling block has reached its maximum height and driller has not changed the direction of travelling block it would try to move further at that movement power would be increased to achieve that target which may caused the failure and error to system. So, operator has to be alert while operating the drawwork system. After simulation based model, prototype model for control the drawworks has been observed. Drawworks system has been observed at forward and backward direction of joystick. Direction of drawworks drum is controlled by joystick, drawworks drum rotates in two ways that is either moving in forward or in backward direction. Joysticks have deadman button it acts as safety features. Stateflow has been used to create the logic for deadman button, if joystick is moved forward or backward to control the drawworks, it would not produce any output signal, until deadman button is not pressed. So deadman button here act as safety feature button for control the drawworks. For ground fault logic has been created in stateflow of joystick that checks ground fault error before and during operation as well. If ground fault is occurred it would stop operating the drawworks. Drawworks has been observed at forward as well as at backward direction using linear and non-linear PI controller at different values of Kp and

Ki to get smooth signal, minimum steady state error, lower rise & fall time and minimum overshoot. When using linear PI controller velocity rise & fall time is lower when it is compared with same parameters when non-linear PI controller is used. But overshoot is greater when linear PI controller is compared with non-linear PI controller, as overshoot is not desireable for system so overall non-linear PI controller is good when it is compared with linear PI controller. Linear and Non-linear PI controller was tuned at various Kp and Ki values desired results for velocity that is achieved at Kp=2.5 and Ki=0.00000025. Slop has been used for joystick to get smooth signal as well as it create deadband region. If their is any minor error in range of ±5% of joystick neutral position drawworks would not be operated. The region where joystick would not give a signal is called deadband region. Weight equilibrium has been done on 14 Kg for travelling block. When weight is more or less than 14 Kg weight equilibrium is distrubed, weight sensor give drive signal to move the travelling block. If weight is lower than a 14 Kg travelling block moves in upward direction and if weight is greater than 14 Kg it moves in downward direction. Weight equilibrium is connected with system by switch so that it can be connect and disconnect according to the system requirement. Weight parameter is one of main parameter during drilling which should be observed carefully because if weight is beyond the capacity it can cause the fault in system, also if load is under weight and system is working on same capacity it cause loss of energy and money. So, by applying these safety features that are deadman button, ground fault detector, deadband region, PI controller tuning and weight equilibrium drawworks system can be prevent from faults and errors, this way safety is increased for drawworks system as well as for driller.

REFERENCES

[1] M. I. Guggari, " Control System For Drawworks ". U.S Patent Patent 6,029,951, 29 Feb. 2000.

[2] "What is weight on bit or WOB ?," [Online]. Available: https://ulterra.com/pdc-bit/weight-on-bit-wob/. [Accessed 3 Feb 2022].

[3] B. Atchison, "Automated well control: A step change in safety," *Digital Chemical Engineering,* vol. 3, 11 March 2022.

[4] G. Lesko, "Electric Drawworks For A Drilling Rig". U.S Patent Patent US 2006/0249719 A1, 9 Nov 2006.

[5] A. G. Dominique Dupuis, "New Remote Control-Conventional Drawwork," in *SPE/IADC Drilling Conference*, Amsterdam, 1995.

[6] V. T. Dipal Patel, "A Review on Casing while Drilling Technology for Oil and Gas Production with Well Control Model and Economical Analysis," *Petroleum,* vol. 5, no. 1, 2018.

[7] I. P. A. Roberto Sanchis, "A new method for experimental tuning of PI controllers based on the step response," *ISA Transactions,* vol. 115, September 2021.

[8] S. Hovda, "Optimal procedures for lowering and applying weight on the bit when starting to drill," *Journal of Petroleum Science and Engineering,* vol. 174, pp. 872-879, March 2019.

[9] "Drilling Manual," 2017. [Online]. Available: https://www.drillingmanual.com/wob-calculations/. [Accessed 13 March 2022].

[10] "Buoyancy," wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Buoyancy#:~:text=Buoyancy%20(%2F%CB%88b%C9%94%C9%AA%C9%99,weight%20of%20the%20overlying%20fluid. [Accessed 13 March 2022].

[11] W. Millauer, "Drive Control With Overload Protection For a Drill Device". U.S Patent Patent 4,885,511, 5 December 1989.

[12] M. F. Mahyari, "The Effects of Drilling Mud and Weight Bit on Stability and Vibration of a Drill String," *Journal of Vibration and Acoustics,* vol. 134, 2012.

[13] H. J. S. Cayeux Eric, "Accuracy and Correction of Hook Load Measurements During Drilling Operations," in *SPE/IADC Drilling Conference and Exhibition*, United Kingdom, 2015.

[14] E. Soeiinah, "Measuring Torque and Hook Load During Drilling". U.S Patent Patent 4,549,431, 29 October 1985.

[15] "What Is MATLAB ?," MathWorks, [Online]. Available: https://www.mathworks.com/discovery/what-is-matlab.html. [Accessed 5 April 2022].

[16] "What Is Simulink ?," MathWorks, [Online]. Available: https://www.mathworks.com/videos/simulink-overview-61216.html. [Accessed 3 December 2021].

[17] "TwinCAT Automation Software," BECKHOFF, [Online]. Available: https://www.beckhoff.com/en-en/products/automation/twincat/. [Accessed 4 December 2021].

[18] "Volt Electric Motor," Voltmotor, [Online]. Available: https://voltmotor.com.tr/downloads/pdf/en/catalogue_en.pdf. [Accessed 27 January 2022].

[19] "TB Load Cell," ESIT, [Online]. Available: https://www.esit.com.tr/en/load_cells/tb__load_cell/urun/176. [Accessed 4 May 2022].

[20] "CX2100-0014 | Power Supply Unit for CX20xx, 130 W," BECKHOFF, [Online]. Available: https://www.beckhoff.com/en-en/products/ipc/embedded-pcs/cx20x3-amd-ryzen/cx2100-0014.html. [Accessed 15 April 2022].

[21] "All-Compatible ACS880 Single Drives," ABB, [Online]. Available: https://new.abb.com/drives/low-voltage-ac/industrial-drives/acs880-single-drives. [Accessed 8 March 2022].

[22] "Avioni," Avioni Technology, [Online]. Available: https://www.avioni.bg/wp-content/uploads/2017/10/AV_JC_CAN.pdf. [Accessed 15 November 2021].