©2014 Acrome Inc., All rights reserved.

Acrome Inc.
ITU ARI3 Science Park
Maslak, Istanbul
Turkey
info@acrome.net
Phone: +90 532 132 17 22
Fax: +90 212 285 25 94

Printed in Maslak, Istanbul

For more information on the solutions Acrome Inc. offers, please visit the web site at:

http://www.acrome.net

This document and the software described in it are provided subject to a license agreement. Neither the software nor this document may be used or copied except as specified under the terms of that license agreement. All rights are reserved and no part may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of Acrome Inc.

ACKNOWLEDGEMENTS

Special thanks to Dr. Murat Yeşiloğlu, Istanbul Technical University, Turkey, and Acrome myControl Team for their supports in embedded control and preparation of the courseware.

Contents

1 Components of Ball Balancing Table 4
1.1 RC Servomotors 4
1.1.1 Overview 4
1.1.2 Theory of Operation 6
1.2 2D Resistive Touch Screen and Controller 7
1.3 Arduino Due 7
1.4 ACROME Power Distribution Box 9
1.5 Mechanics of Ball Balancing Table 9
2 Pulse Width Modulation and RC Servomotor 10
2.1 Understanding Pulse Width Modulation (PWM) 10
2.1.1 In-Lab Exercise 11
2.2 Driving an RC Servomotor with PWM 11
3 Derivative Filtering 13
3.1 In-Lab Exercise 13
4 System Modeling 15
4.1 Non-Linear Equation of Motion 17
4.1.1 Lagrangian Method 17
4.1.2 Newton's Law of Motion 19
4.2 Modeling of Ball Balancing Table System 20
4.2.1 Linearization around Operating Point 22
4.2.2 Obtaining Transfer Function of Plant 22
4.3 Modeling of Actuator 23
4.4 Cascade Control of Ball Balancing Table 23
5 Performance Measures 25
5.1 Understanding Percentage Overshoot, Settling Time, Peak Time and Steady State Error 25
5.1.1 Damping Ratio (ξ) 25
5.1.2 Natural Frequency (ω_n) 25
5.1.3 Percentage Overshoot 25
5.1.4 Peak Time (t_p) 27
5.1.5 Settling Time (t_s) 27
5.1.6 Steady State Error 27
5.1.7 In-Lab Exercise 28
6 Control System Design 28
6.1 Academic PID controller 29
6.2 Parallel PID controller 29

6.3 Root Locus 30
6.4 P Controller 30
6.4.1 Sample Design with P Controller 31
6.4.2 In-Lab Exercise 34
6.5 PD Controller 36
6.5.1 Sample Design with PD controller 36
6.5.2 In-Lab Exercises 39
6.6 PV Controller 40
6.6.1 Sample Design with PV controller 41
6.6.2 In-Lab Exercises 43
6.7 PID Controller 45
6.7.1 Sample Design with PID controller 46
6.7.2 In-Lab Exercises 48
7 Frequency Response Analysis 50
7.1 Introduction 50
7.2 Frequency Response of a Linear System 51
7.3 In Lab Exercise: Frequency Response of BBT 52
References 56

1 Components of Ball Balancing Table

The main parts of "ACROME Ball Balancing Table" can be seen below.

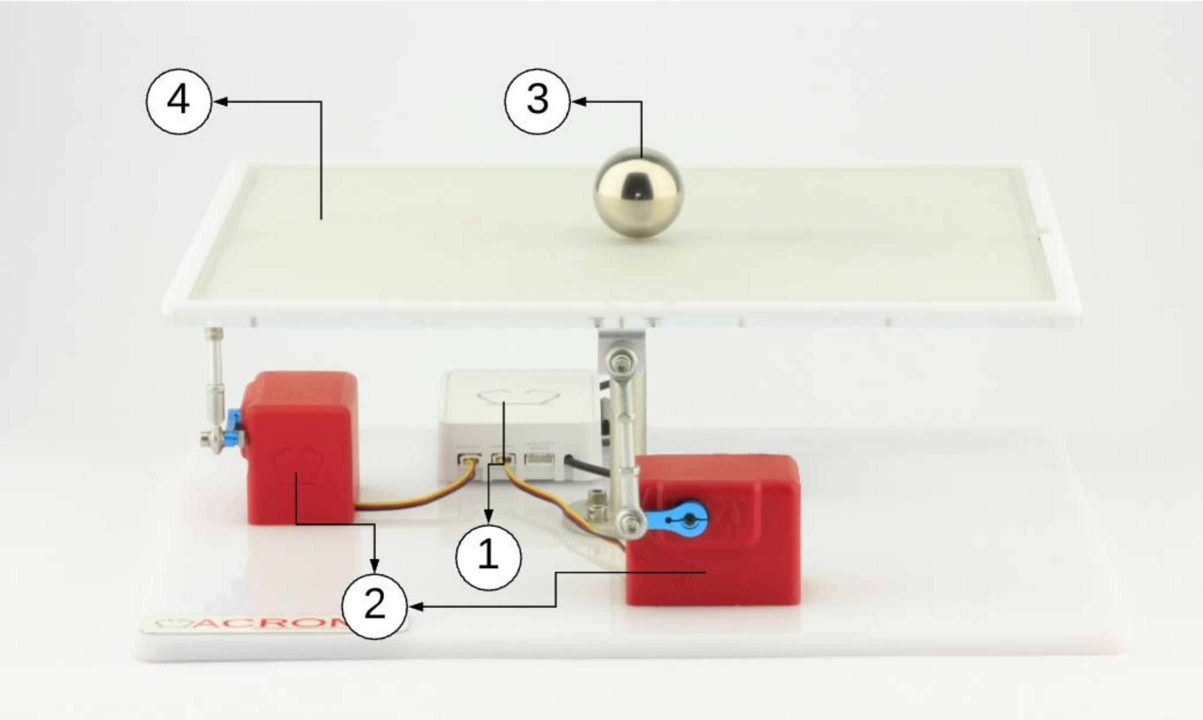


Figure 1.1: Components: 1. Power Distribution Box 2. RC Servos 3. Steel Ball

4. 2D Resistive Touch Sensor and Controller

1.1 RC Servomotors

1.1.1 Overview

RC servos are electromechanical devices that convert electrical signals to rotary movement. They provide simple and handy solutions to most of control and robotic applications.

Figure 1.2: RC Servomotor

RC servomotors consist of several parts that include:

Controller: Circuit that is responsible of reading the control signals (PWM signals –more about PWM signals can be found in Chapter 2) and controlling the motor accordingly. Controller circuits determine the type of servos; they can be either digital or analog. Analog servos handle PWM signals up to 50 Hz frequency. Digital servos handle these PWM signals more precisely. They can decode signals up to 330 Hz. This difference in decoding can provide more torque to the motors. In addition to that; digital servos can be programmed to change direction, dead band width etc. Generally; digital servos are more advantageous than analog servos in expense of cost and power consumption.

Potentiometer: Position feedback of the main shaft is provided by the potentiometer. It is attached to the driveshaft so the rotation of the driveshaft results in different resistances on the potentiometer. By reading the resistance values, controller knows exact driveshaft angle.

Motor: The motors of the servos are commonly high speed DC motors which are controlled by H-bridges located inside their controller circuitry.

Gearbox: The gear set is located between the driveshaft and the motor. It regulates the RPM of the motor resulting in lower movement speed and more torque.

Driveshaft: Driveshaft is the output of the whole system. It is the actual component that rotates to the desired angle.

Connector: They generally have three pins that carry “+”, “-“ and “signal” to the controller. Connectors may have different color codes depending on the manufacturer.

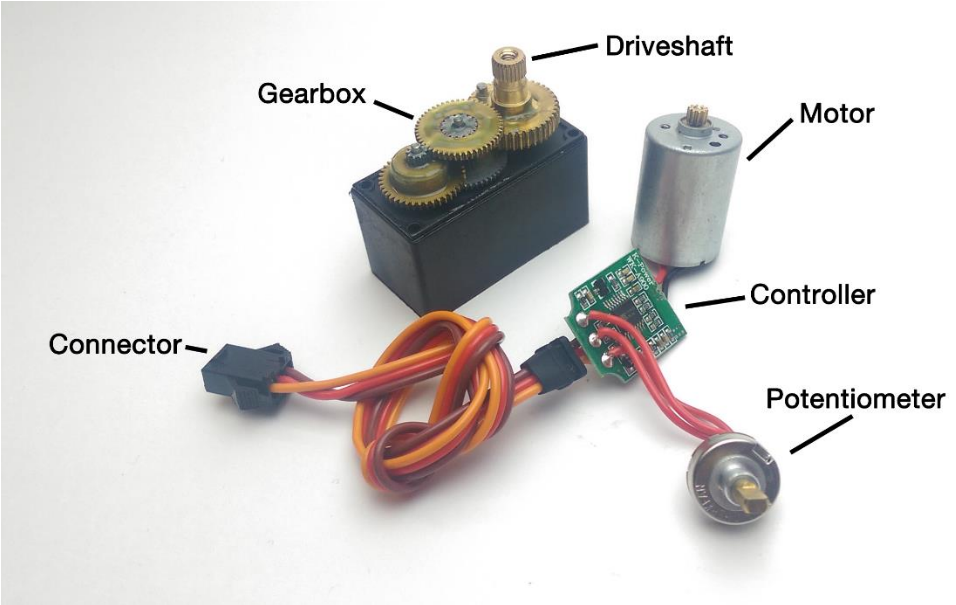


Figure 1.3: RC Servo Components

1.1.2 Theory of Operation

The servo is actually a closed loop control system that requires continuous input signal. Servo operation can be explained with few basic steps:

1. Controller decodes the PWM input signal and converts it into a voltage that corresponds to an angle.
2. Controller reads the potentiometer voltage values and determines the shaft position.
3. Controller calculates the error from the difference between input and potentiometer voltage.
4. Controller converts error into the H-bridge output.

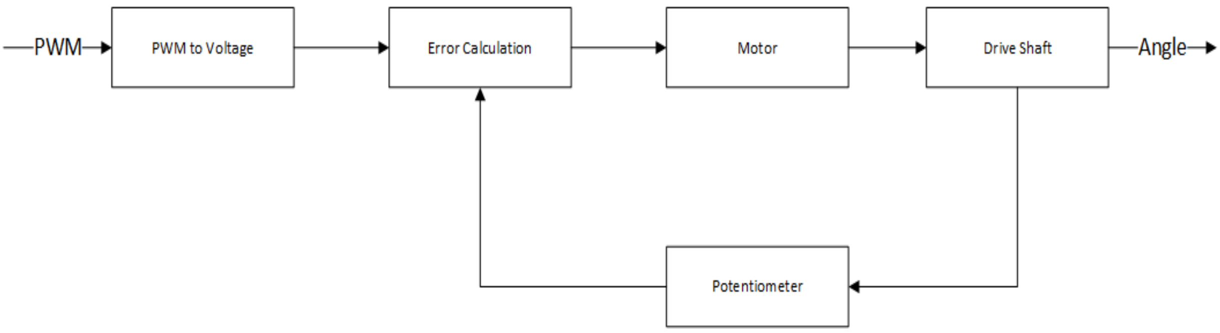


Figure 1.4: RC Servo Theory of Operation

1.2 2D Resistive Touch Screen and Controller

Position feedback of the ball on the table is acquired through 17-inch, 4 wires touch screen. Basically, a touch screen consists of two sheets that have a voltage gradient produced by resistors on them. These sheets are separated by an air gap. When two sheets are pressed together, voltage divided on each sheet is read by the touchscreen controller that translates these voltages to screen coordinates.

The touch screen located on the ball balancing table is capable of delivering up to 100 Hz position data via a digital controller that enables much smoother feedback acquisition.

1.3 Arduino Due

Arduino is a portable embedded device that allows users to design and control robotic or mechatronic systems.

Figure 1.7: Arduino Due

The Arduino Due is the first Arduino board based on a 32-bit ARM core microcontroller. With 54 digital input/output pins, 12 analog inputs, it is the perfect board for powerful larger scale robotic projects.

The board contains everything needed to support the microcontroller; simply connect it to a computer with a micro-USB cable or power it with an AC-to-DC adapter or battery to get started. The Due is compatible with all Arduino shields that work at 3.3V and are compliant with the 1.0 Arduino pinout.

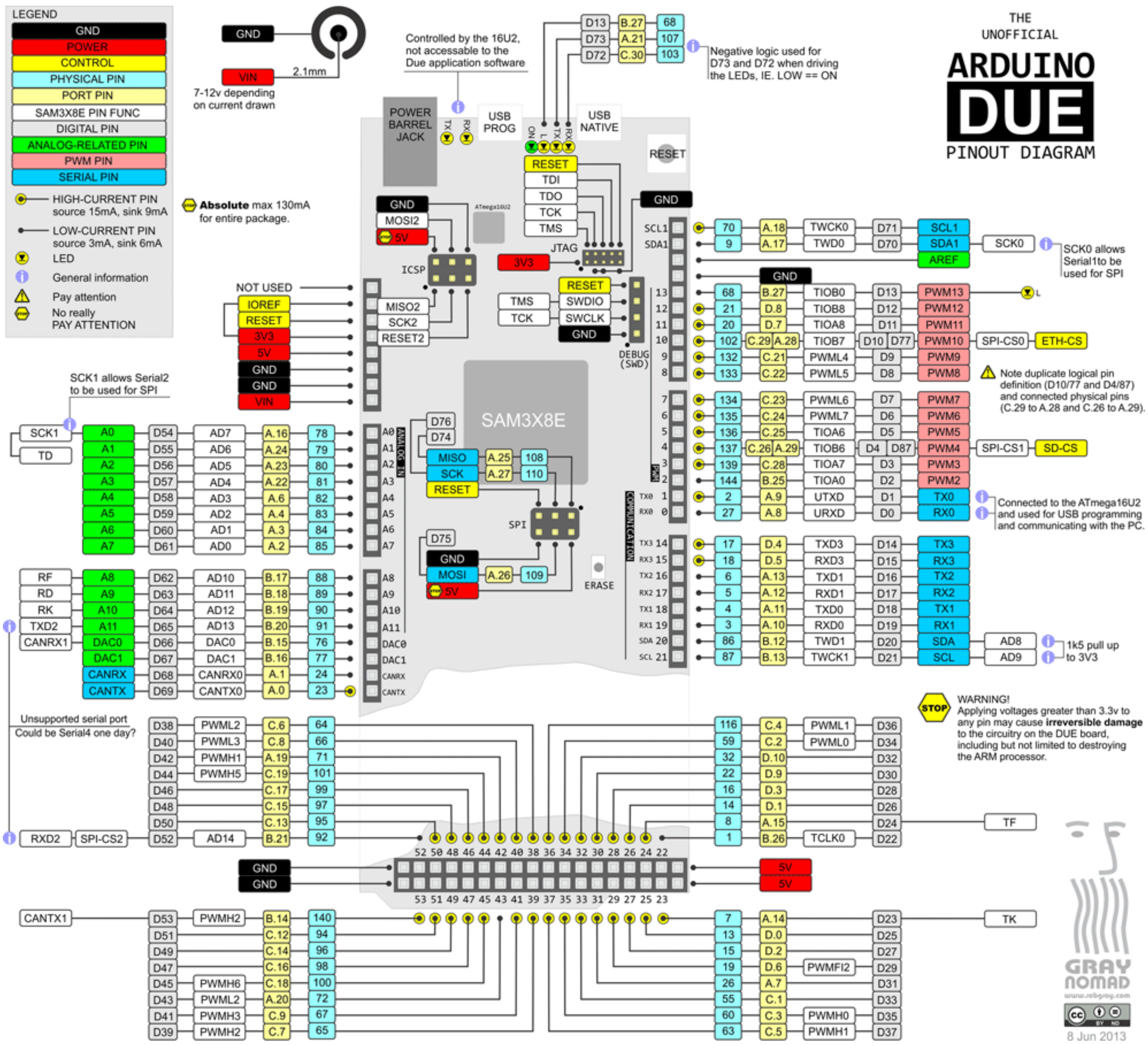


Figure 1.8: Arduino Ports and Peripherals

1.4 ACROME Power Distribution Box

Two RC servomotors and Arduino connections are located on ACROME power distribution box which is shown in Figure 1.7. It has also RGB led and switch mode regulators.

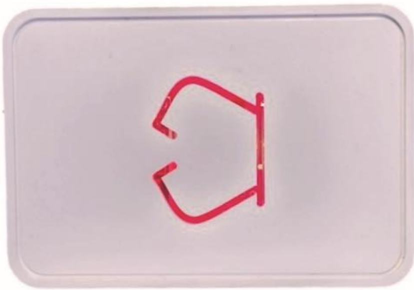


Figure 1.7: ACROME Power Distribution Box

1.5 Mechanics of Ball Balancing Table

Table plane moves freely along two axes with rotary joints. Each servo is connected to this plane and delivers their actions via rod end bearings.

2 Pulse Width Modulation and RC Servomotor

Pulse Width Modulation Technique is mainly used in industry. The technique produces analog signals by using digital signals. It can be easily produced and applied. As for the RC servomotor, it uses simple on/off pulsed signal. With respect to its input signal, RC servomotor gives an angle as an output. As a result, Pulse Width Modulation and RC Servomotor complement each other well.

2.1 Understanding Pulse Width Modulation (PWM)

Pulse Width Modulation (PWM) is a solid method to encode analog signals digitally by generating a square wave with a certain duty cycle that is modulated to encode the analog signal. PWM technique is widely used in various applications such as switching power supplies and motor control.

PWM technique produces a square PWM signal that has several properties like period, frequency, amplitude and duty cycle. Period is the time interval required to repeat the signal and frequency is the number occurrences of these signals in 1 second. Amplitude shows the voltage level of the high or low state of the PWM signal. Duty cycle is used to describe high state of the signal as a percentage of the total period and finally, pulse width is the time that signal is at the high state.

A typical PWM signal and its properties are shown in the Figure 2.1:

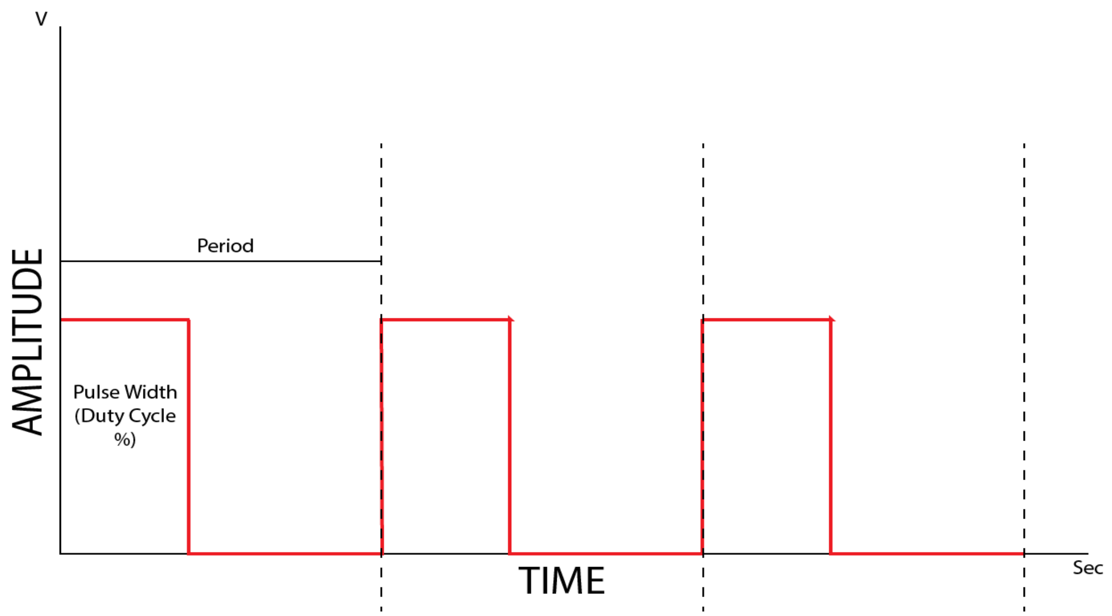


Figure 2.1: Typical PWM Signal with its Properties

2.1.1 In-Lab Exercise

1. Open "GeneratingandReadingPWM.slx" where Figure 2.2 shows its front panel.

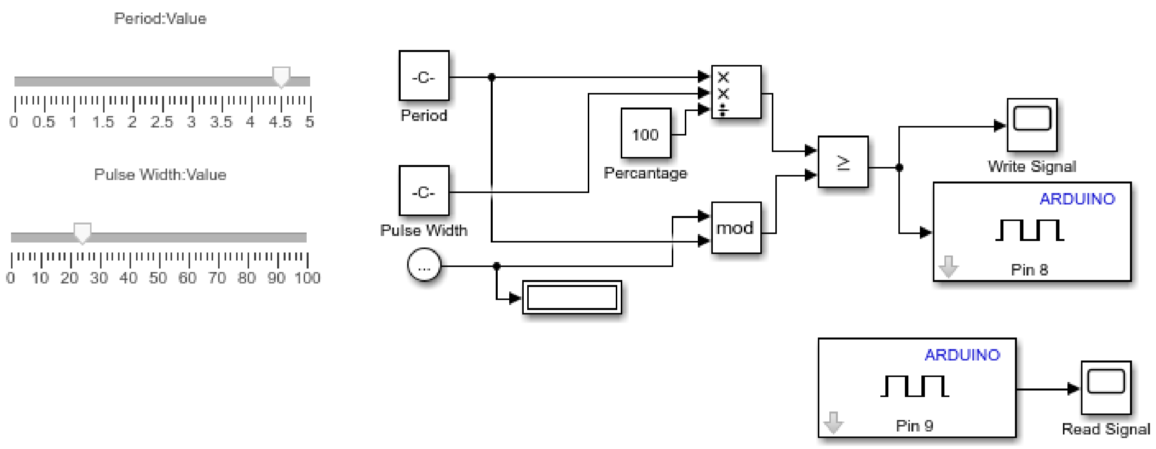


Figure 2.2: Front Panel of "GeneratingandReadingPWM.slx"

2. Change the percentage duty cycle and the period as desired,
3. Run the program, use a jumper cable to short-circuit between pins D8 and D9.
4. Double click and open Read Signal and Write Signal Scope. Observe simulated PWM from the oscilloscope,
5. Compare the consistency of the performance obtained from sliders and scopes.

2.2 Driving an RC Servomotor with PWM

RC servos can be either analog or digital. Type of the servomotor affects the properties of the required signal to control servos. All analog servos have common similar signals that have 20 millisecond (ms) periods, therefore 50 Hz frequencies. Duty cycle end time or Pulse Width is between 600 and 2400 microseconds (µs). Pulse width determines the position of the servos. Usually; 600 µs refers to -90° position, 1500 µs is neutral or 0° position and 2400 µs refers to +90° position. Changes in these times are linear so other desired time values for various positions can be easily calculated. It should be noted that these values may not correspond with other servos. Relation between pulse width and the position may vary.

Amplitude variation does not affect the servo control too much as long as servo control circuit decodes and understands position. Many microcontrollers have output voltage levels that change between 3.3V and 5V that could be decoded by integrated circuits of servos.

PWM signals that are required to control an analog RC servomotor in relationship with its position can be seen in Figure 2.3.

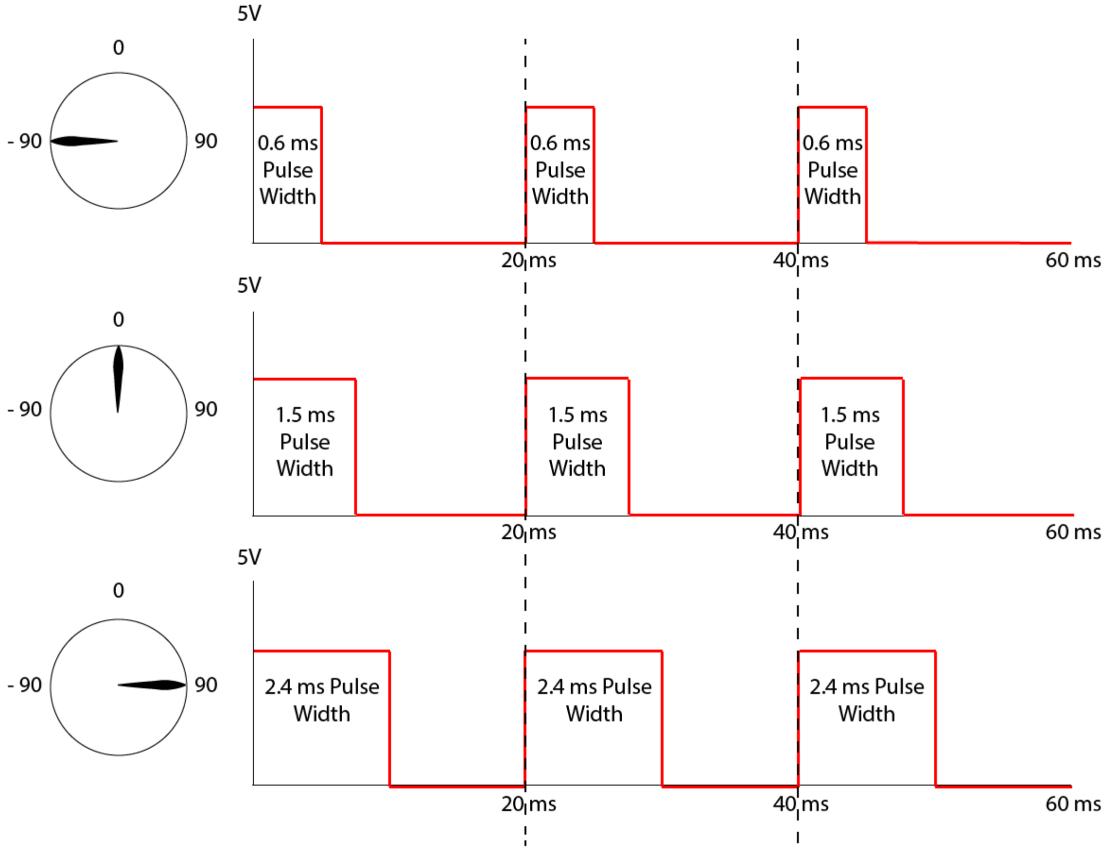


Figure 2.3: PWM Signal and Servomotor Position Relation

3 Derivative Filtering

There are two main reasons of derivative ripples: measurement noise and discrete time derivative. While reading the ball position from the touch screen, high frequency noises and nonlinear effects may distort the measured data. As a result, there is a difference between the actual and the measured position arising from measurement noise.

In addition, the loop time is a few multiple of a millisecond, even small changes in measured data are divided by the loop time, so the derivative of them is a high number and this causes many ripples. Therefore, an internal filter must be used to make the system response slower and to reduce ripples. A low pass filter is applied to the derivative term in the PID controller as an internal filter. A cut off frequency of "30" has been chosen as the internal filter through trial and error. The transfer function of this filter is shown below:

$$Filter(s) = \frac{30}{s + 30}$$

3.1 In-Lab Exercise

1. Open the "DerivativeFiltering.slx",

2. You will see the following front panel and the block diagram shown below:

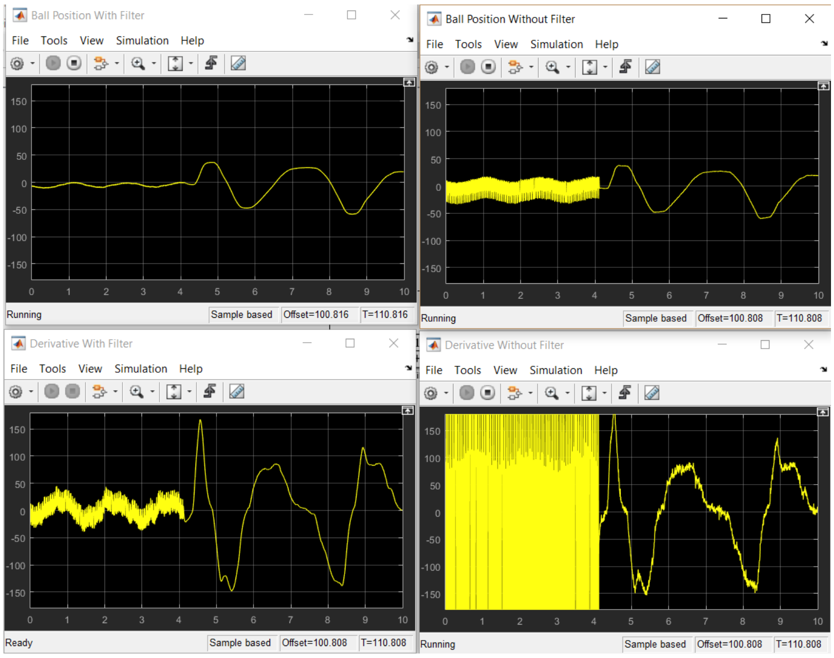


Figure 3.1: Front Panel of "DerivativeFiltering.slx"

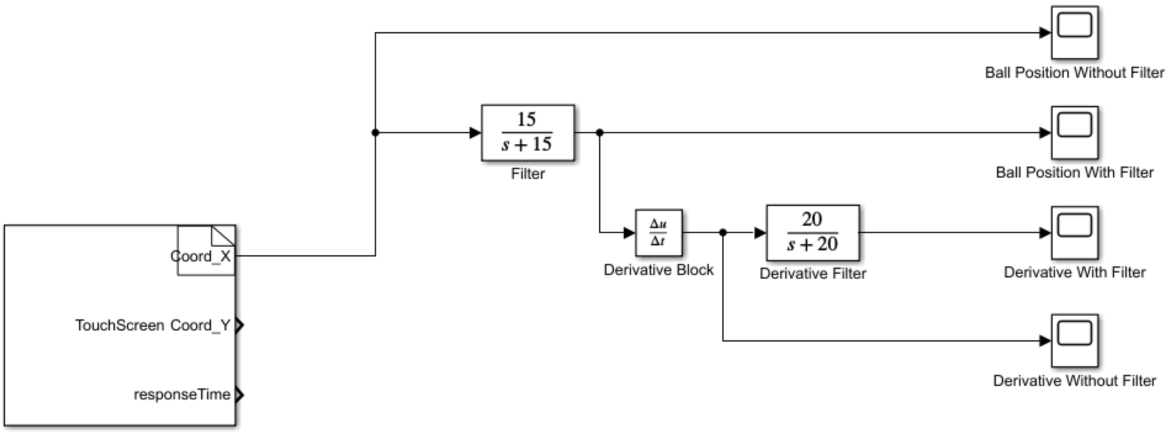


Figure 3.2: Block Diagram of "Derivative Filtering.slx"

In order to observe the filter effects, there must be a position input on the touchscreen. Real time position of the ball can be seen on the scopes. The data points show the real position of the ball. Non-Filtered and Filtered Derived Position Data charts can be seen in the scopes. Finally, filter parameters can be seen in Figure 3.2. It is pre-configured to the optimal value mentioned above.

3. Touch the panel and notice that the data points on the charts are moving along with your touch. This is your position feedback. You can also try this with the ball.
4. While touching the panel, observe the charts. As you can see the non-filtered derived position data has more spikes or ripples that need to be ruled out before any calculation. Notice that the filter eliminates the non-filtered data from spikes and ripples.
5. Go ahead and change the filter parameters. How will the filter affect the system with different parameters in terms of noise, amplitude and phase?
6. Describe the advantage and disadvantage of using a filter in control applications.

4 System Modeling

The physical model of the Ball Balancing Table is shown in Figure 4.1. The model describes the kinematic system of the Ball Balancing Table. According to the physical model, the actuators of the system (motor x and motor y) are attached rigidly to the base plate of the system. The actuators are connected to the table over links with ball joints on both sides. Table is connected to the base plate of the system over a rotary joint with two degree of freedom. The physical model also illustrates the linkage parameters of the table which are necessary for the mathematical model of the system. These parameters are also explained below:

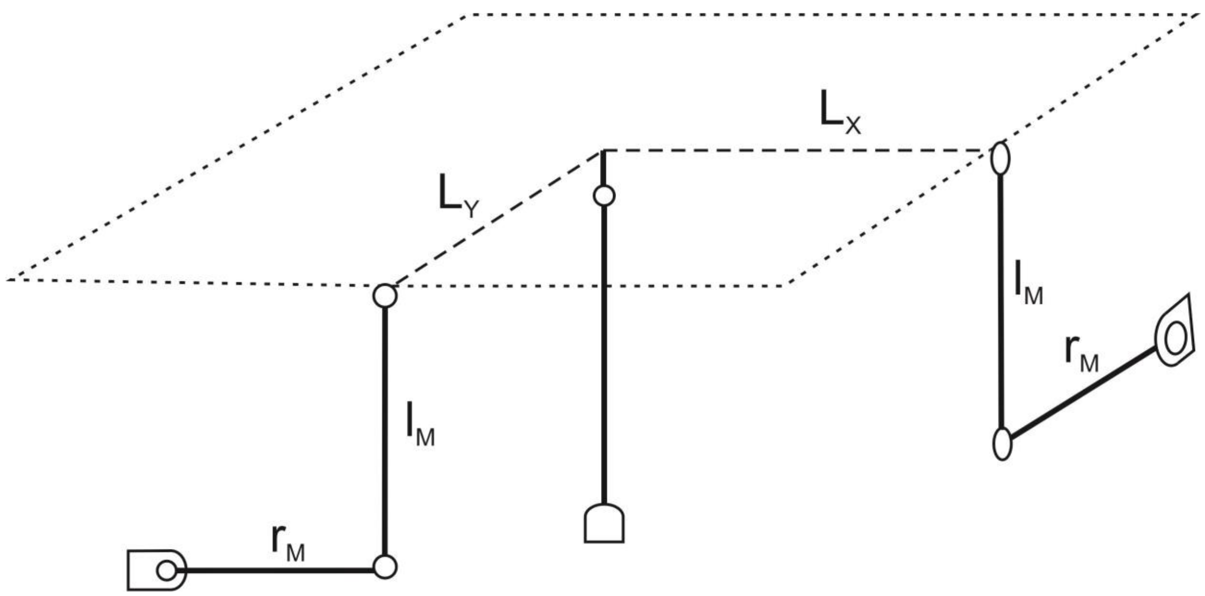


Figure 4.1: The Physical Model of the Ball Balancing Table

Table 4.1: Ball Balancing Table Parameters

Symbols	Definition	Value	Unit
L_x	Beam Length in x-direction	0.134	[m]
L_y	Beam Length in y-direction	0.168	[m]
r_M	Motor Arm Length	0.0245	[m]
r_b	Radius of the Ball	0.02	[m]
m_b	Mass of the Ball	0.26	[kg]
J_b	Rotational Inertia of the Ball	0.0000416	[kg*m^2]
g	Acceleration due to Gravity	9.81	[m/s^2]
α	Plate Angle about x-axis	(Variable)	[degrees]
β	Plate Angle about y-axis	(Variable)	[degrees]
ϑ_x	Motor Angle for x-axis	(Variable)	[degrees]
ϑ_y	Motor Angle for y-axis	(Variable)	[degrees]

Before obtaining the model of the system, the following assumptions must be considered:

The model of the system assumes that the ball-plate contact is not lost under any circumstances. Another important assumption is that the ball rolls on the table without slipping. To simplify the model, all friction forces and resulting torques are neglected.

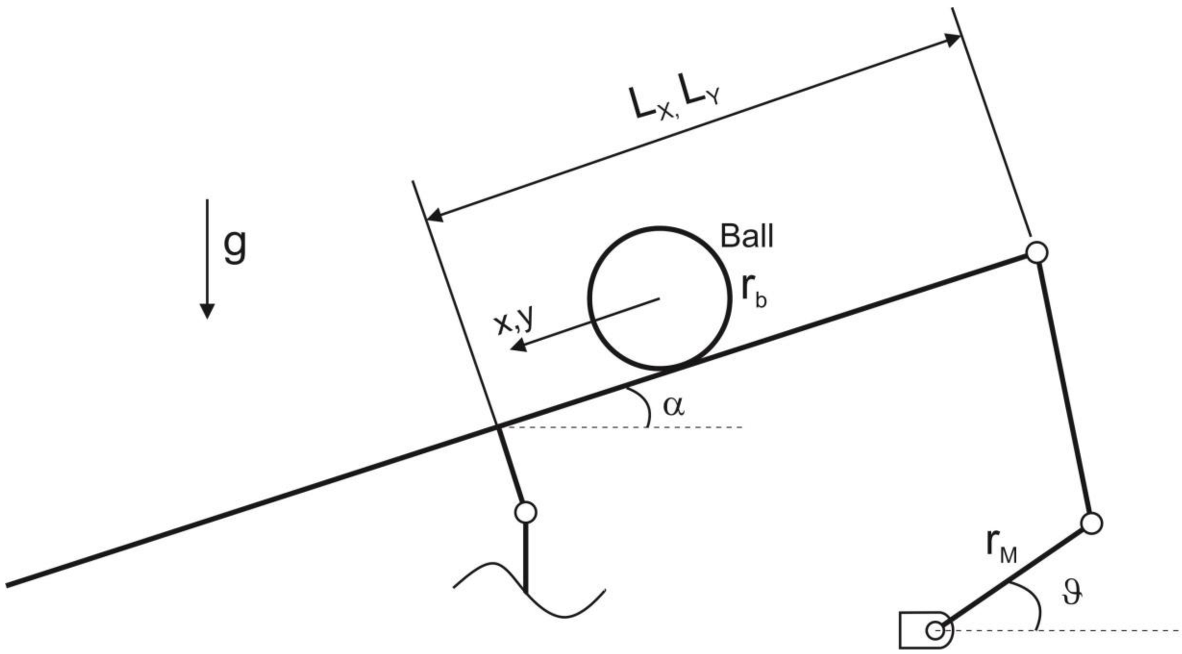


Figure 4.2: Free Body Diagram

4.1 Non-Linear Equation of Motion

The first step of modeling is to derive the equation of motion. The equation of motion describes the relationship between motion of the ball (x, y) and the two angles of the table (α, β). In the following sections, the equation of motion will be derived with two different methods.

4.1.1 Lagrangian Method

The Lagrangian Method derives the equation of motion through the relation of kinetic and potential energy of the system. The Lagrangian Method is a useful method for complex systems which generally have more degrees of freedom than one. The Lagrangian Equation:

$$\frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = 0 \tag{4.1}$$

L (Lagrangian) is the difference between the kinetic and potential energy of the system.

$$L = E_{kin} - E_{pot} \tag{4.2}$$

To derive the equation of motion, the kinetic and the potential energy of the system must be obtained. The total kinetic energy of a rolling ball (without slipping) can be described as:

$$E_{kin} = E_{kin,T} + E_{kin,R} = \frac{1}{2} m_b v_b^2 + \frac{1}{2} J_b \omega_b^2 \tag{4.3}$$

$E_{kin,T}$ is the translational kinetic energy and $E_{kin,R}$ is the rotational kinetic energy of the system.

$$E_{kin,T} = \frac{1}{2} m_b v_b^2 = \frac{1}{2} m_b (\dot{x}_b^2 + \dot{y}_b^2) \tag{4.4}$$

$$E_{kin,R} = \frac{1}{2} J_b \omega_b^2 = \frac{1}{2} J_b \frac{v_b^2}{r_b^2} = \frac{1}{2} J_b \frac{(\dot{x}_b^2 + \dot{y}_b^2)}{r_b^2} \tag{4.4}$$

Thus,

$$E_{kin} = \frac{1}{2} m_b (\dot{x}_b^2 + \dot{y}_b^2) + \frac{1}{2} J_b \frac{(\dot{x}_b^2 + \dot{y}_b^2)}{r_b^2} = \frac{1}{2}\left(m_b + \frac{J_b}{r_b^2}\right)(\dot{x}_b^2 + \dot{y}_b^2) \tag{4.6}$$

The potential energy of the ball for given plate angles can be described as:

$$E_{pot} = m_b g h_b = -m_b g x_b \sin(\alpha) - m_b g y_b \sin(\beta) \tag{4.7}$$

Therefore, the Lagrangian is equal to:

$$L = E_{kin} - E_{pot} = \frac{1}{2}\left(m_b + \frac{J_b}{r_b^2}\right)(\dot{x}_b^2 + \dot{y}_b^2) + m_b g x_b \sin(\alpha) + m_b g y_b \sin(\beta) \tag{4.8}$$

The partial derivative terms of the Equation 4.1 for the x-direction are as follows:

$$\frac{\partial}{\partial t}\left(\frac{\partial L}{\partial \dot{x}}\right) = \frac{\partial}{\partial t}\left(\left(m_b + \frac{J_b}{r_b^2}\right)\dot{x}_b\right) = \left(m_b + \frac{J_b}{r_b^2}\right)\ddot{x}_b \tag{4.9}$$

And

$$\frac{\partial L}{\partial x} = m_b g \sin(\alpha) \tag{4.10}$$

So, the differential equation of the motion for the x-direction becomes:

$$\left(m_b + \frac{J_b}{r_b^2}\right)\ddot{x}_b - m_b g \sin(\alpha) = 0 \tag{4.11}$$

Solving the equation for $\ddot{x}_b$, the required form of the equation is derived:

$$\ddot{x}_b = \frac{m_b g r_b^2}{m_b r_b^2 + J_b} \sin(\alpha) \tag{4.12}$$

Applying the same method to the y-direction, the equation of motion for y-direction can be derived:

$$\ddot{y}_b = \frac{m_b g r_b^2}{m_b r_b^2 + J_b} \sin(\beta) \tag{4.13}$$

4.1.2 Newton's Law of Motion

To apply the Newton's Law of Motion, the system should be decoupled into two modes of motion (motion in x and y direction). Considering the motion in x-direction, according to the Newton's Law of Motion, the sum of forces acting on the ball equals:

$$\sum F = m_b \frac{d^2 x_b}{dt^2} \tag{4.14}$$

Assuming there is no friction and viscous damping in the system, the forces acting on the ball can be described as:

$$\sum F = F_{x,t} - F_{x,r} \tag{4.15}$$

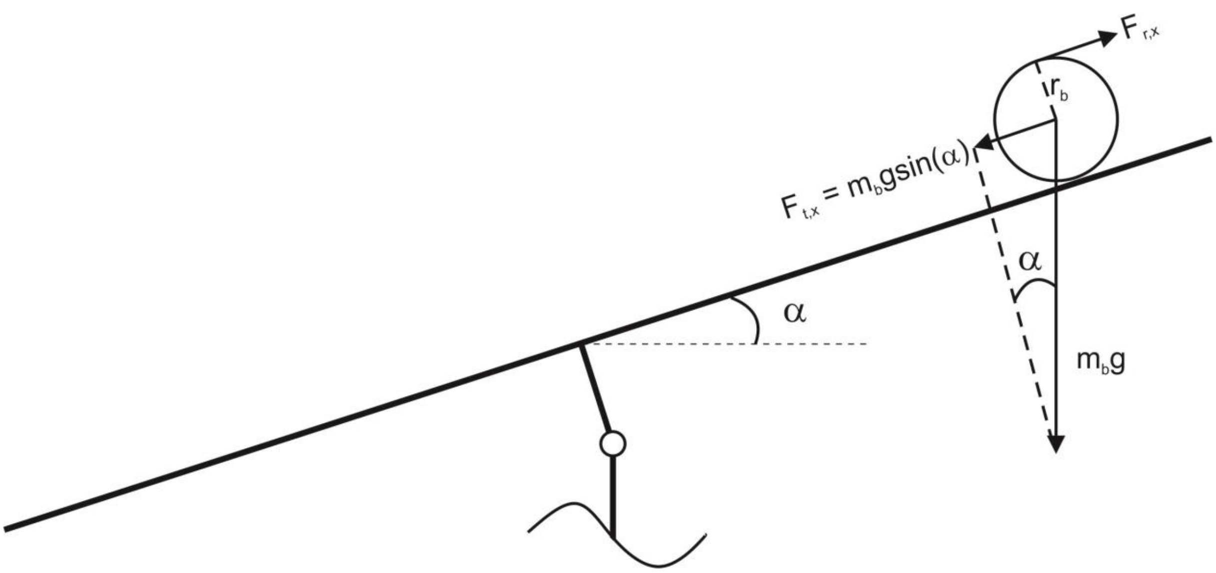


Figure 4.3: Free Body Diagram

Where $F_{x,t}$ is the translational force acting on the ball due to gravity and inclined plane as shown in Figure 4.3.

$$F_{x,t} = mg\sin(\alpha) \tag{4.16}$$

$F_{x,r}$ is the force acting on the ball due to the ball's rotation. The torque generated due to the ball's rotation is:

$$T = F_{x,r} r_b = \frac{J_b}{r_b}\ddot{x}_b \longrightarrow F_{x,r} = \frac{J_b}{r_b^2}\ddot{x}_b \tag{4.17}$$

Thus,

$$m_b \ddot{x}_b = m_b g\sin(\alpha) - \frac{J_b}{r_b^2}\ddot{x}_b \tag{4.18}$$

By solving the equation for $\ddot{x}_b$, the required form of the equation is derived:

$$\ddot{x}_b = \frac{m_b g r_b^2}{m_b r_b^2 + J_b}\sin(\alpha) \tag{4.19}$$

Applying the same method to the y-direction, the equation of motion for y-direction can be derived:

$$\ddot{y}_b = \frac{m_b g r_b^2}{m_b r_b^2 + J_b}\sin(\beta) \tag{4.20}$$

As it can be seen in the Equation (4.1) and (4.18), the derived equations of motion for both methods are same, if the same assumptions are made before the derivation.

4.2 Modeling of Ball Balancing Table System

The aim of modeling is to derive the physical relationship between the outputs and inputs of a system. In our case, the outputs of the system are the coordinates of the ball in x and y-direction (x_b, y_b). The inputs of the system are the motor angles $(\vartheta_x, \vartheta_y)$ as it can be seen in Figure 4.4.

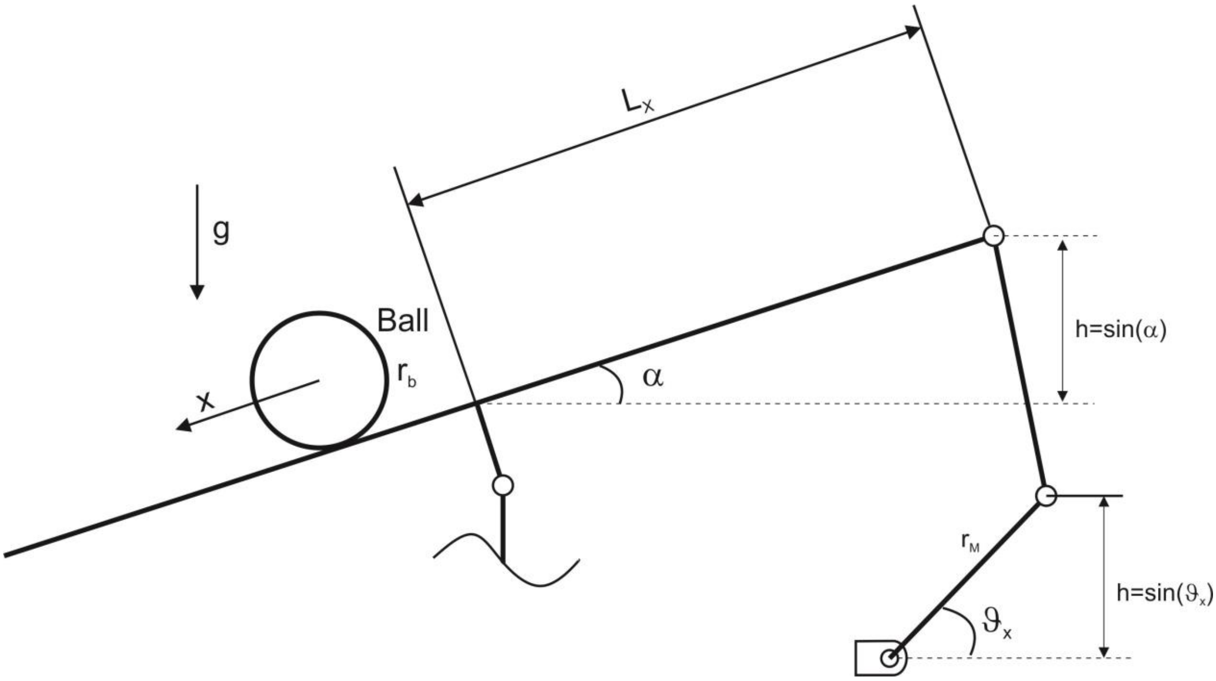


Figure 4.4: Relationship between Motor Angles and Plate Angles

Figure 4.4 also illustrates that:

$$\sin(\vartheta_x)r_M = \sin(\alpha)L_x = h \tag{4.21}$$

Thus, the following relationship between plate angle and motor angle can be obtained:

$$\sin(\alpha) = \frac{r_M}{L_x}\sin(\vartheta_x) \tag{4.22}$$

Also, for the y-direction:

$$\sin(\beta) = \frac{r_M}{L_y}\sin(\vartheta_y) \tag{4.23}$$

We can write instead of $\sin(\alpha)$ and $\sin(\beta)$ terms in the Equations 4.19 and 4.20 the right side of the Equations 4.22 and 4.23. Consequently, we have the differential equations of the system which describe the relationship between the outputs and inputs of the system:

$$\ddot{x}_b = \frac{m_b g r_b^2 r_M}{(m_b r_b^2 + J_b)L_x}\sin(\vartheta_x) \tag{4.24}$$

$$\ddot{y}_b = \frac{m_b g r_b^2 r_M}{(m_b r_b^2 + J_b)L_y}\sin(\vartheta_y) \tag{4.25}$$

4.2.1 Linearization around Operating Point

To obtain the transfer function of the system, the differential equations must be linearized about the operating point (*x=0, y=0*). For small angles:

$$\sin(\vartheta_x) \approx \vartheta_x, \quad \sin(\vartheta_y) \approx \vartheta_y \tag{4.26}$$

Thus Equations 4.24 and 4.25 can be re-written as:

$$\ddot{x}_b = \frac{m_b g r_b^2 r_M}{(m_b r_b^2 + J_b) L_x} \vartheta_x = K_{BBT,X} \vartheta_x \tag{4.27}$$

$$\ddot{y}_b = \frac{m_b g r_b^2 r_M}{(m_b r_b^2 + J_b) L_y} \vartheta_y = K_{BBT,Y} \vartheta_y \tag{4.28}$$

By using the value of system parameters which were listed in Table 4.1, $K_{BBT,X}$ and $K_{BBT,Y}$ can be easily calculated as 1 and 1.255 respectively.

4.2.2 Obtaining Transfer Function of Plant

As it is seen in Equations 4.27 and 4.28, the systems variables are decoupled so that we can investigate every mode of motion separately.

The plant's differential equation in x-direction is:

$$\ddot{x}_b = K_{BBT,X} \vartheta_x \tag{4.29}$$

Taking the Laplace transform we obtain the equation below:

$$s^2 X_b(s) = K_{BBT,X} \vartheta_x(s) \tag{4.30}$$

$$G_{BBT,X}(s) = \frac{X_b(s)}{\vartheta_x(s)} = \frac{K_{BBT,X}}{s^2} = \frac{1}{s^2} \tag{4.31}$$

The transfer function in y-direction can also be derived the Equation 4.28:

$$G_{BBT,Y}(s) = \frac{X_b(s)}{\vartheta_y(s)} = \frac{K_{BBT,Y}}{s^2} = \frac{1.255}{s^2} \tag{4.32}$$

4.3 Modeling of Actuator

The actuator of ball balancing table is a servomotor. The servomotor's transfer function can be approximated as a first order function in the form as below:

$$G_M(s) = \frac{\vartheta(s)}{V_m(s)} = \frac{K_M}{\tau s + 1} \tag{4.33}$$

K_M is the gain and τ is the time constant of the actuator. K_M is calculated as 100 and τ is calculated as 0.01. After the calculations, motor transfer function is obtained as below:

$$G_M(s) = \frac{100}{0.01s + 1} \tag{4.34}$$

4.4 Cascade Control of Ball Balancing Table

Ball balancing table has a cascade structure which includes an inner and an outer loop as shown in Figure 4.5.

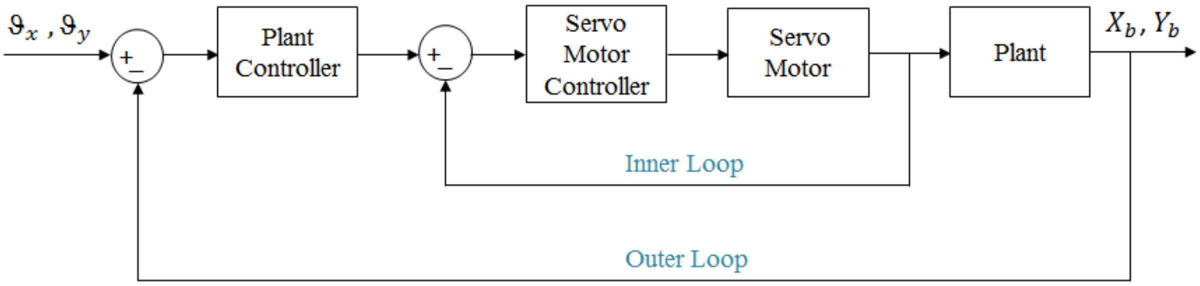


Figure 4.5: Block Diagram of Ball Balancing Table

Inner loop is a unity feedback closed loop system and controls the servomotor position. Servomotor of the ball balancing table automatically controls its position thanks to its own controller which is positioned under the touch screen. Inner loop reaches the steady state in an instant. Therefore, inner loop transfer function can be obtained as a first order transfer function. When the inner loop is reduced, the new block diagram is as below:

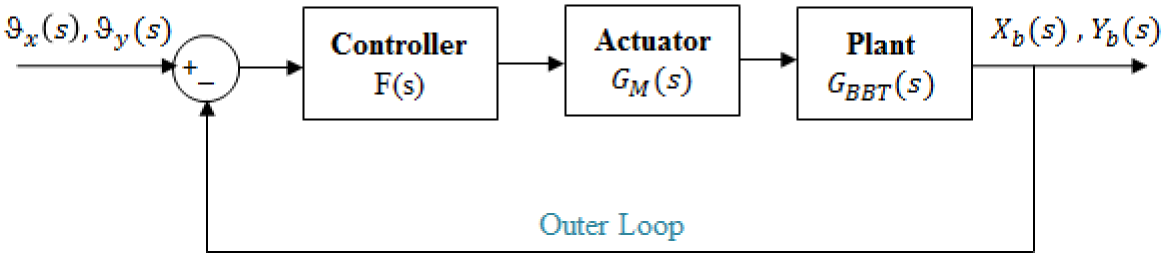


Figure 4.6: Block Diagram of System after Reduction

Plant's and Actuator's transfer functions were obtained in the previous section. When the parameters are replaced, block diagram for x and y-direction is also shown as below:

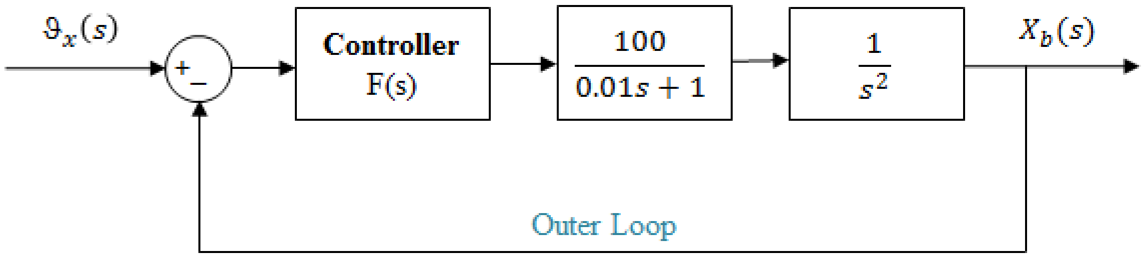


Figure 4.7: Block Diagram of Outer Loop in x-direction

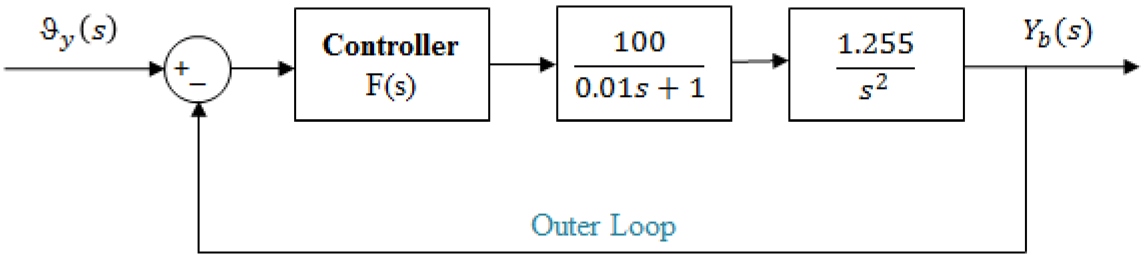


Figure 4.8: Block Diagram of Outer Loop in y-direction

The outer loop consists of a plant, an actuator and their controller. Designing the controller will be explained in Chapter 6.

5 Performance Measures

Performance measures or criteria are defined for second order systems via applying a step input. Different performance measures are determined by considering the dynamics of systems. Depending on performance measures, damping ratio (ξ) and natural frequency (ω_n) are found using formulas in Chapter 5.1. Also, if damping ratio and natural frequency are known, performance measures can be calculated. In this section, all parameters are investigated.

5.1 Understanding Percentage Overshoot, Settling Time, Peak Time and Steady State Error

5.1.1 Damping Ratio (ξ)

When step input is applied, the system has an overshoot depending on the damping ratio. Damping ratio and the system response coupling can be seen in Table 5.1. By using percentage overshoot (PO), damping ratio can be calculated. The relation between damping ratio and percentage overshoot can be seen below:

$$\xi = \frac{-\ln(PO/100)}{\sqrt{\pi^2 + (\ln(PO/100))^2}}$$

5.1.2 Natural Frequency (ω_n)

While damping ratio equals to “0”, natural frequency is the frequency of oscillation of the system. Natural frequency is calculated using settling time or peak time.

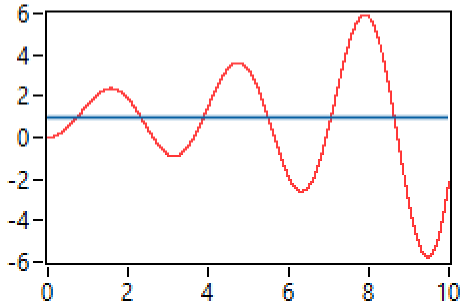
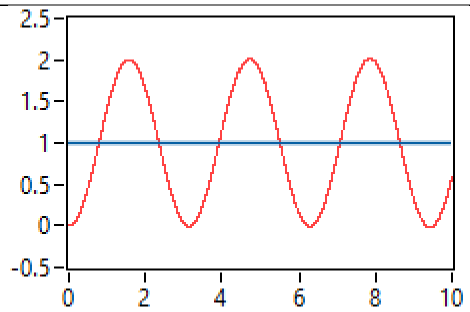
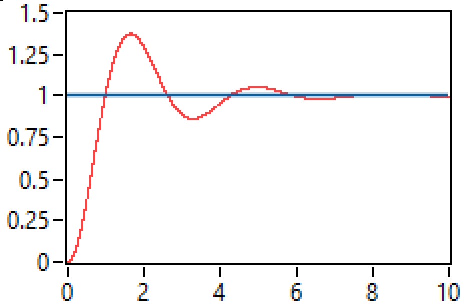
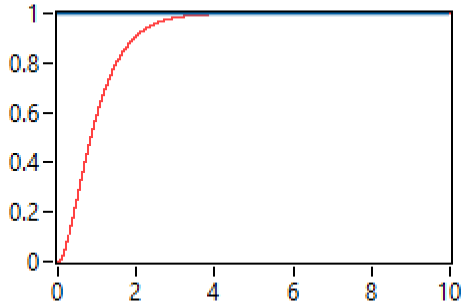
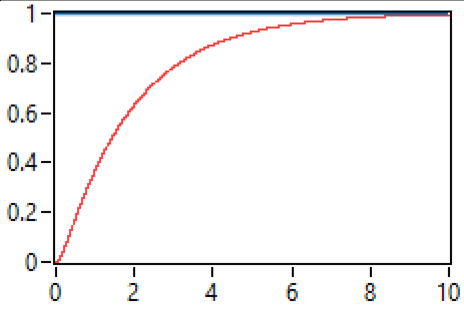
5.1.3 Percentage Overshoot

For a stable system, overshoot is difference between maximum and final output values. Percentage overshoot is 100 multiplied by ratio of overshoot to final output value. Also, percentage overshoot can be calculated from the step input response of the system, but if damping ratio is known, other formula of percentage overshoot can be used.

$$Percentage\ Overshoot = 100 * \frac{Maximum\ Output\ Value - Final\ Output\ Value}{Final\ Output\ Value}$$

$$= 100 * e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}}$$

Table 5.1: Damping Ratio, System Behaviour and System Response Graph

Damping Ratio (ξ)	System Behaviour	System Input and Output
$\xi<0$	Unstable	
$\xi=0$	Undamped	
$0<\xi<1$	Underdamped	
$\xi=1$	Critically Damped	
$1<\xi$	Over damped	

5.1.4 Peak Time (t_p)

Peak time is the elapsed time from applying the step input to the maximum output value. For overdamped systems, the peak time is not defined. Also, peak time can be calculated using the following formula, which includes the damping ratio and the natural frequency.

$$t_p = \frac{\pi}{\omega_n * \sqrt{1-\xi^2}}$$

5.1.5 Settling Time (t_s)

Settling time is defined when the response remains within 2% or 5% of the final value. 2% band of the final value will be used for subsequent calculations. Also, there is a formula for calculation of settling time below:

$$t_s = \frac{4}{\xi * \omega_n}$$

5.1.6 Steady State Error

Steady state error is the difference between the reference input and the final output value. Most of the time, it is an undesirable situation where the system response does not settle to reference input and it is defined as steady state error. If the system has a steady state error, it can be removed easily by adding an integral type of controller.

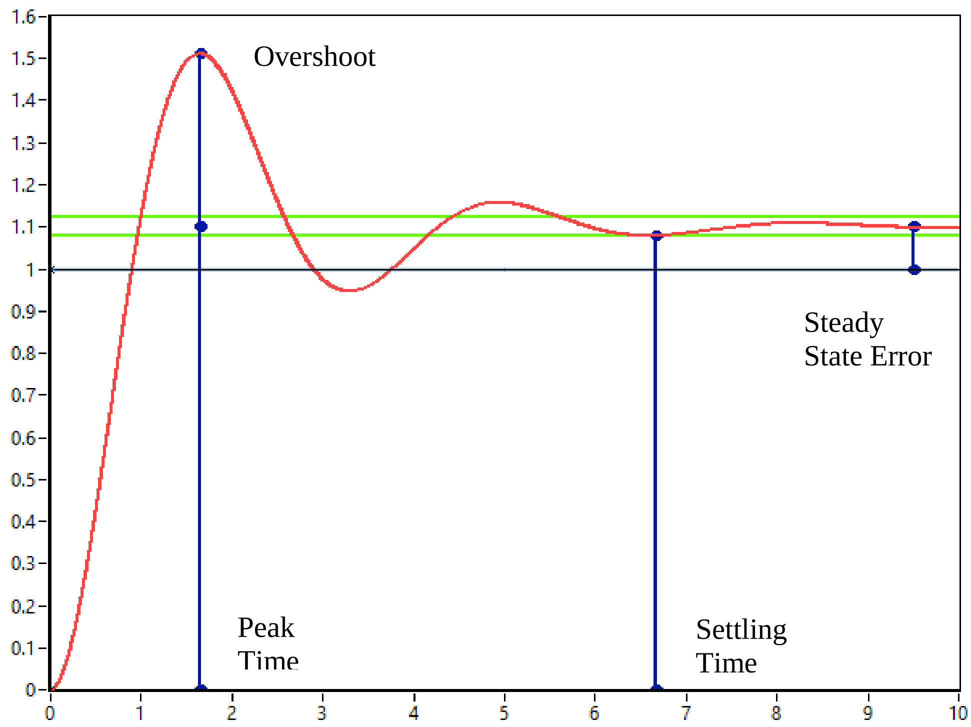


Figure 5.1: Overshoot, Settling Time, Peak Time and Steady State Error

A second order system's general transfer function is below:

$$\frac{K\omega_n^2}{s^2 + 2\xi\omega_n s + \omega_n^2}$$

System response final value is the limit of the transfer function as "s" goes to "0". With respect to the transfer function above, final value is "K" for unit step input.

5.1.7 In-Lab Exercise

1. Open "PerformanceMeasures.slx"

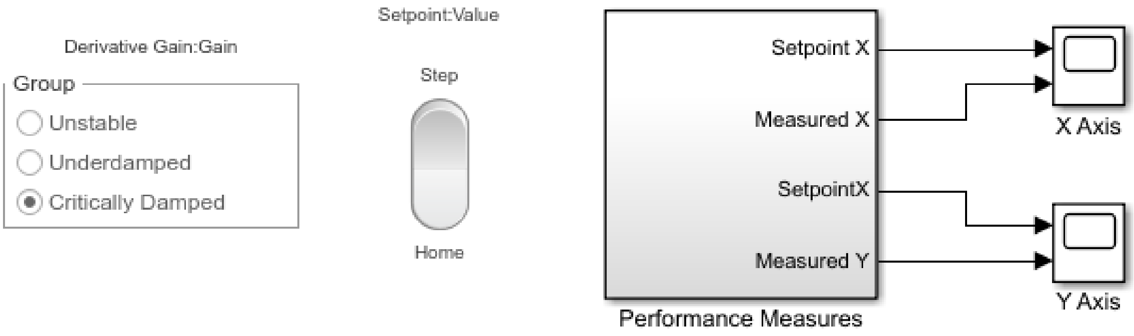


Figure 5.1.7: Front Panel of "PerformanceMeasures.slx"

2. Run the program. Select one option from Unstable, Underdamped, Critically Damped and click Step from switch.
3. Calculate the following performance measures from graph: percentage overshoot, settling time, peak time and steady state error.
4. Observe the system input and output on the graph and determine the performance measures.
5. Click Home from switch and select a different option. Click Step then observe the changes in performance measures.

6 Control System Design

Open loop systems are called the manual control systems since the output has no effect on the input. Although open loop structure is simple, economic and stable; it is inaccurate, unreliable and has no disturbance rejection. Closed loop systems are called automatic control systems and they have many advantages like disturbance rejection, noise attenuation, presence of nonlinearity, and robustness; so closed loop control systems are preferred in control applications generally.

Achieving the previously specified performance measures is the purpose of the controller design with closed loop system. Choosing the type of the controller depends on system objectives and needs. In general, there are three tuning parameters; proportional (P), integral (I) and derivative (D). P-term depends on the present error, I-term accumulates the past errors and D-term predicts the future errors.

All classic linear controllers are based on combinations of these three terms on forward and feedback paths. There are many types of linear controller such as P, PI, PD, PID, PI-PD, PV etc. PID controller is called three-mode controller and it is the most common control algorithm used in industry. In general, when derivative term adjusts the transient response, integral term eliminates the steady state error. There are three major structures of PID algorithms: academic, parallel and serial form. Academic and serial forms of PID controller are described as interacting algorithms. On the other hand, parallel form is described as a non-interacting algorithm. Academic and parallel forms of PID controllers and fuzzy controller will be used for ball balancing table's position control. Also, other types of controller such as P, PI and PD are obtained by using academic and parallel forms of PID.

6.1 Academic PID controller

Academic PID controller is mathematically expressed as:

$$u(t) = K_c\left(e(t) + \frac{1}{T_i}\int e(t)dt + T_d\frac{de(t)}{dt}\right)$$

The proportional term is the gain of the controller so integral and derivative actions are dependent on the value of K_c. K_c is real and has a finite value; T_d is the derivative term time constant and T_i is the integral term time constant.

6.2 Parallel PID controller

Parallel PID controller is mathematically expressed as:

$$u(t) = K_p e(t) + K_i\int e(t)dt + K_d\frac{de(t)}{dt}$$

Proportional term is independent of integral and derivative terms. Three tuning parameters of parallel PID controller are determined separately. K_p, K_d *and* K_i are real and have finite values similar to the gain of the academic PID controller.

Academic PID controller form can be converted into the parallel form by using basic algebraic calculations. The inverse of this conversion is also possible.

6.3 Root Locus

Stability of the system can be determined directly from its closed loop poles. In order to achieve this, W.R. Evans developed a graphical method called root locus to determine locations of all possible closed loop poles in s-domain with respect to gain (K).

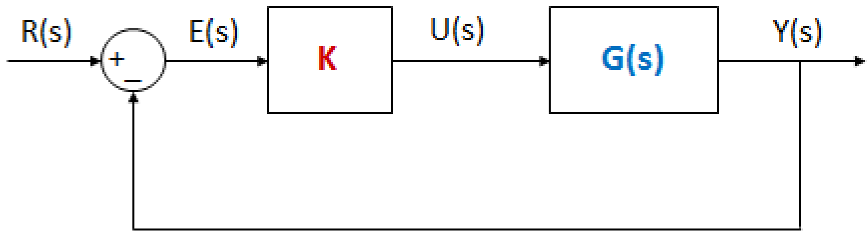


Figure 6.1: General Block Diagram of Root Locus

To draw a root locus, firstly open loop transfer function has to be obtained. Poles and zeros of the open loop system are located on the complex plane. Secondly, open loop system is multiplied by a proportional gain. Then the loop is closed with a unity feedback path. Denominator of the closed loop transfer function is described as a characteristic polynomial. Characteristic polynomial is a function of K as shown below:

$$C(s) = 1 + KG(s)$$

Finally, locations of the closed loop poles are calculated by solving the characteristic equation for each value of K. When all these poles are drawn on the complex plane, root locus is ready to be analyzed. Root locus is also used for controller design, but our controller will not be designed using the root locus.

6.4 P Controller

P type controller is the simplest controller and can be easily implemented to a system. In addition, it does not change the order of the system. Closed loop system with P controller satisfies only one performance measure and only one closed loop pole is located as desired.

Therefore, it is not sufficient to have more than one desired criteria. P controller does not eliminate steady state errors. With P controller, controller gain increases, and the error is smaller but high increase in gain can cause a high increase in control signal, overshoot or even can cause instability of the system.

When the integral and derivative terms are zero, academic PID controller becomes P-type controller in a similar way. If the integral time constant is infinite and the derivative time constant is zero, parallel PID controller becomes P-type controller. General block diagram of the closed loop system with P type controller is below:

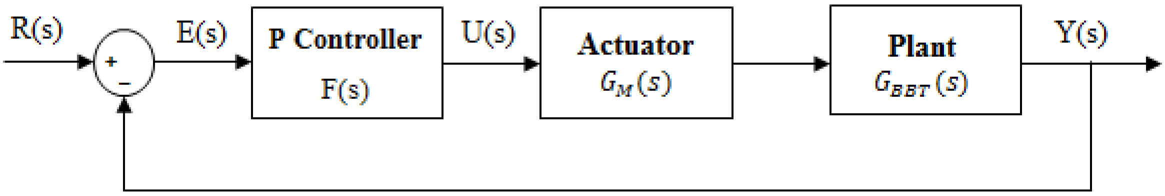


Figure 6.2: General Block Diagram with P Controller

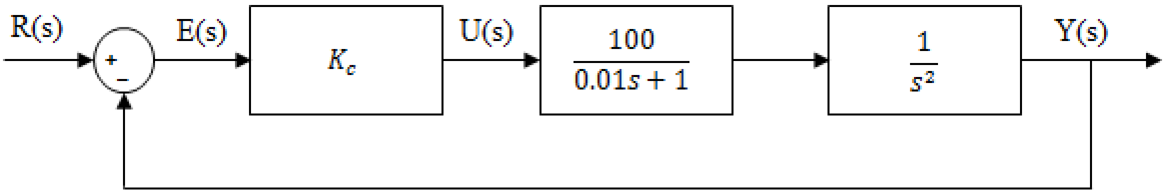


Figure 6.3: The Closed Loop System with P Controller in x-direction

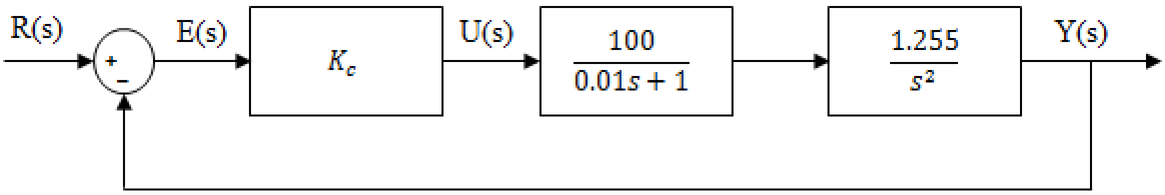


Figure 6.4: The Closed Loop System with P Controller in y-direction

6.4.1 Sample Design with P Controller

We will design the controller in y-direction. Transfer functions of the controller, the actuator and the plant are below respectively:

$$F(s) = K_c, \qquad G_M(s) = \frac{100}{0.01s+1}, \qquad G_y(s) = \frac{1.255}{s^2}$$

We obtain the closed loop transfer function depending on Figure 6.4.

$$T_y(s) = \frac{Y(s)}{R(s)} = \frac{F(s) * G_M(s) * G_{BBT,y}(s)}{1 + F(s) * G_M(s) * G_{BBT,y}(s)} = \frac{125K_c}{0.01s^3 + s^2 + 125K_c}$$

Designed characteristic polynomial is the denominator of the closed loop transfer function shown below:

$$P_c(s) = 0.01s^3 + s^2 + 125K_c$$

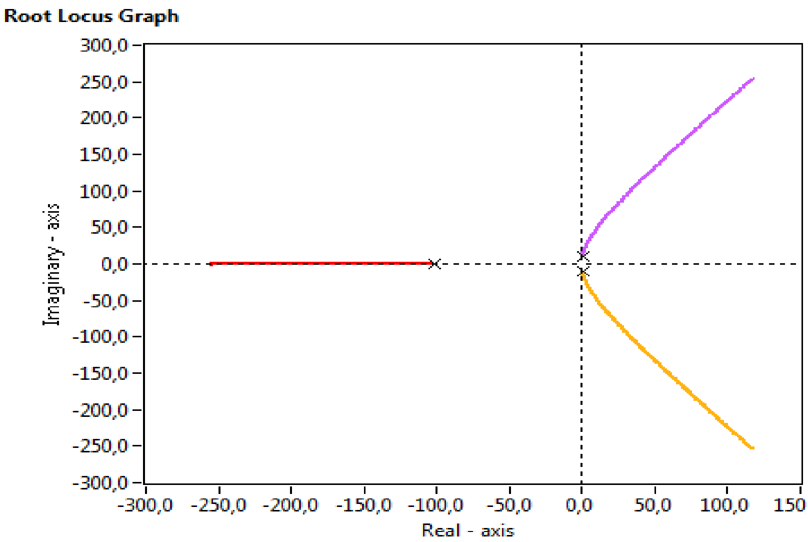


Figure 6.5: Root Locus Graph of Closed Loop System with P Controller

The root locus of the open loop system is drawn in Figure 6.5. If we analyze the root locus, there are three poles of the closed loop system; a complex conjugate pole pair and one real pole. For a linear system to be stable, all poles must have negative real parts. Real part of the complex conjugate pole pair is positive for each value of K_c so all poles are located in the right half plane. As a result, we expect to find that the closed system unstable. In other words, the system's output is approaching to infinity as time increase. The expected step response of the closed loop system is shown for $K_c = 0.25$ below:

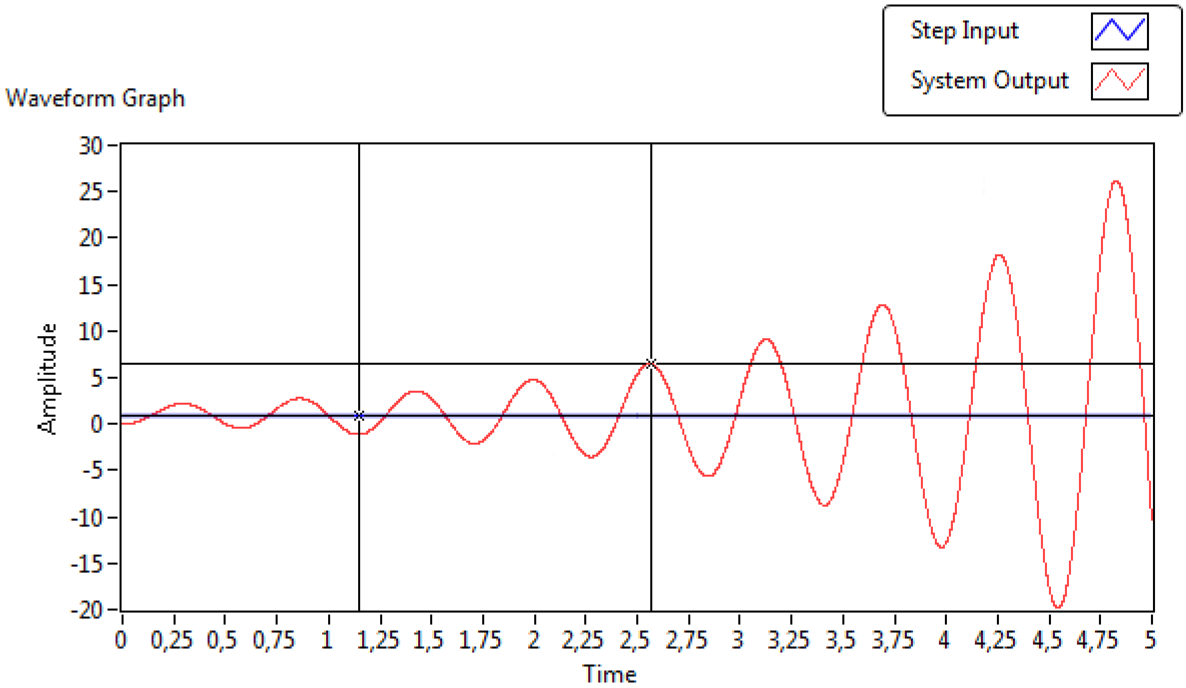


Figure 6.6: The Unit Step Response for Simulation with P Controller on Y

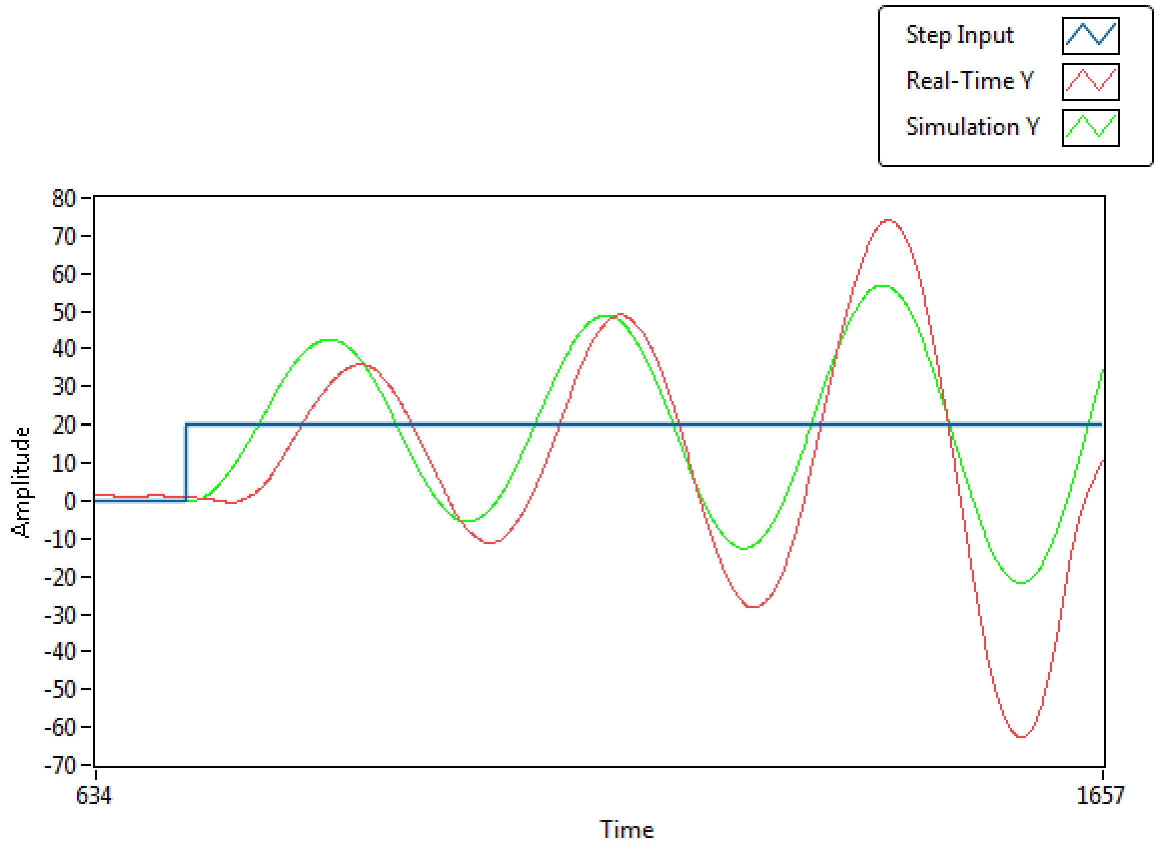


Figure 6.7: The Real-Time, Simulation and Step Input Graphs with P Controller on Y

6.4.2 In-Lab Exercise

6.4.2.1 Control & Simulation Loop Configuration

Before the exercise, some configurations are necessary for Model Configuration Parameters. Same configurations will be applied to all Model Configuration Parameters.

We must click to Model Configuration Parameter and select “Solver” selection. Configurations in popup window will be in the Figure 6.8 shown below. Finally, we will click to the “OK” button.

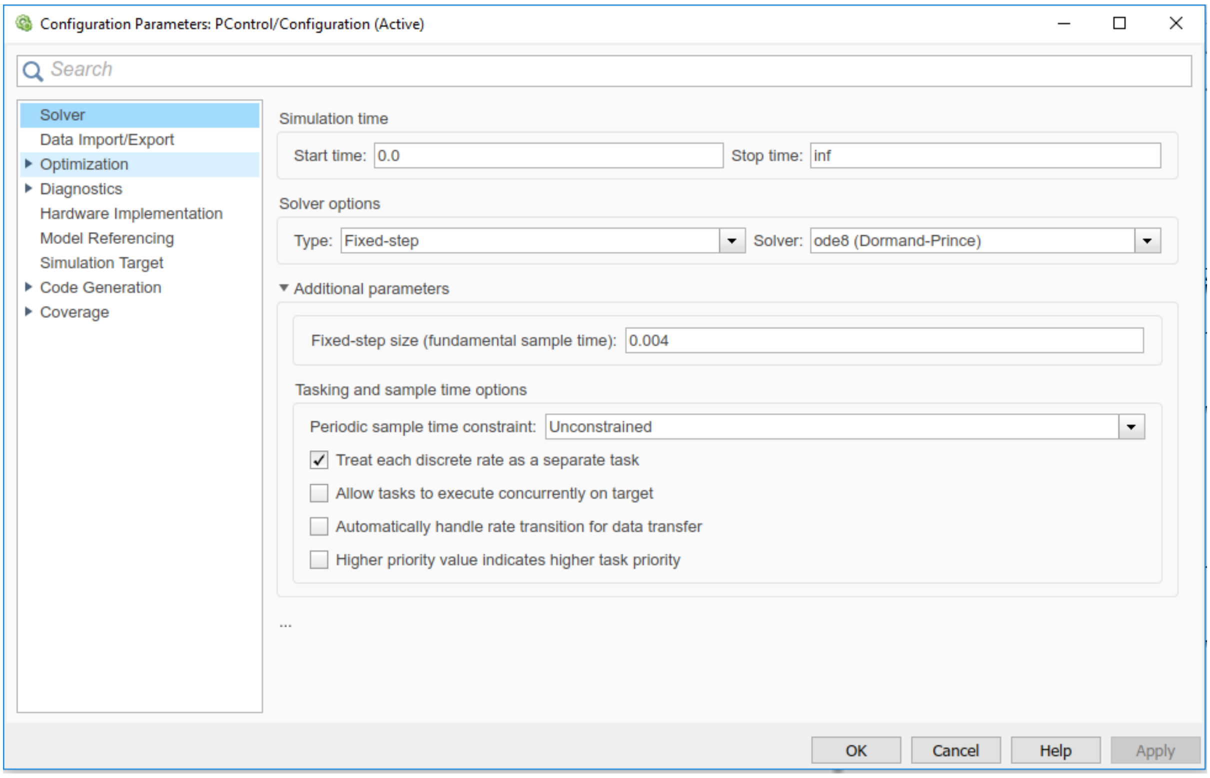


Figure 6.8: Configuration Simulation Parameters…

6.4.2.2 In-Lab Exercise

1. Open the “PControl.slx”,
2. You will see the following block diagram shown below:

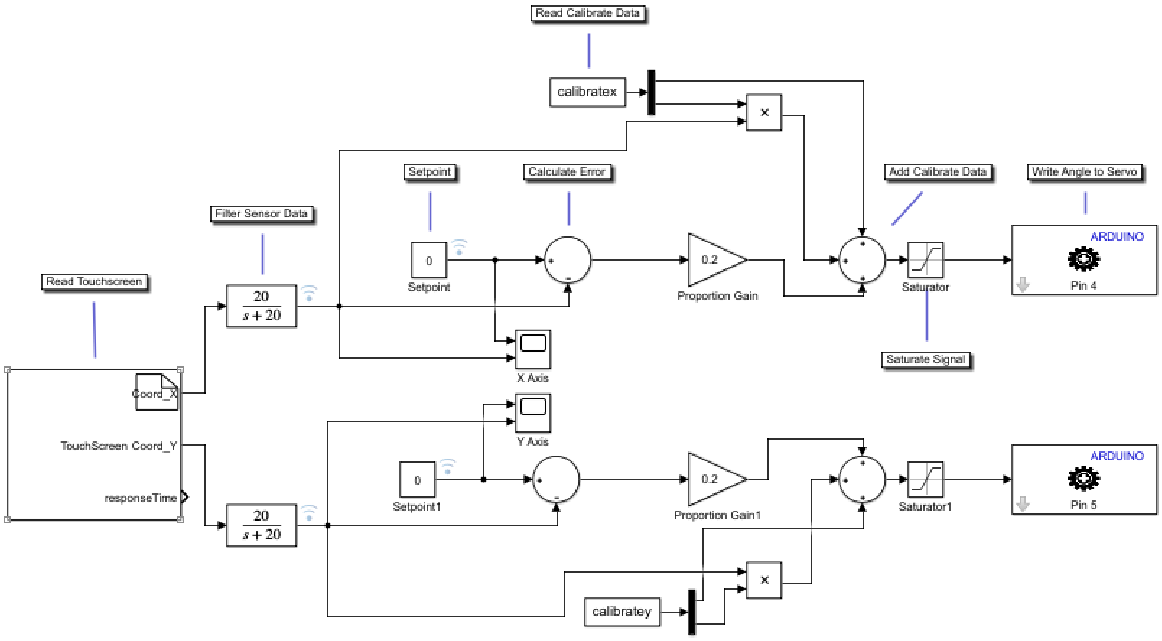


Figure 6.4.2.2: Block Diagram of "PControl.slx"

3. Run the program. Double click to scope and open.
4. Observe real system responses.

Answer the following questions for the block diagram as shown in Figure 6.3. Sample design of P controller is for y-direction. We expect to solve In-Lab Exercise for the x-direction. Use "BBT P Controller Design" for this example.

1. Determine the order of the open loop system without a controller (Hint: Multiplication of the plant and the motor transfer function gives the open loop system transfer function).
2. Find the closed loop transfer function of the system with P controller.
3. Obtain the characteristic polynomial of the closed loop system.
4. What is the order of the closed loop system? Does it change?
5. Draw the root locus of the system. What do you expect the response of the closed loop system will be if a unit step input is applied?
6. Is the closed loop system stable? If not, can it be stable?

6.5 PD Controller

PD controller is generally used for position control applications because of the plant's integrator. It approaches two closed loop system poles to desired locations and adds a zero at the same time without changing the order of the system. PD controller adds one zero to the system. When the step input is applied at the beginning, derivative of the initial error approaches infinite in an instant. This pulse is described as a derivative kick. Design of the PD controller on the forward path results a derivative kick. The other disadvantage of PD controller is that it cannot reject disturbance and eliminate the steady state error. Closed loop system block diagrams with PD controller can be seen as follows:

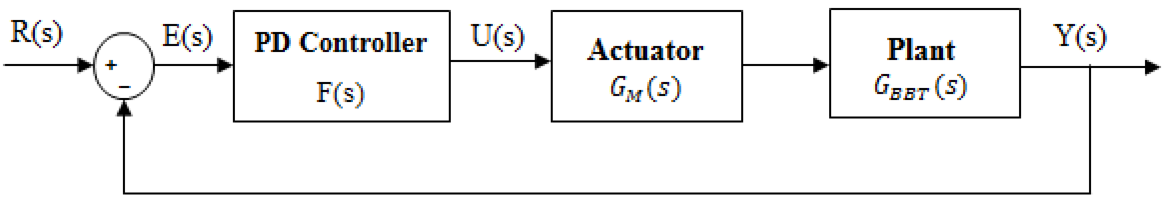


Figure 6.9: General Block Diagram of the Closed Loop System with PD Controller

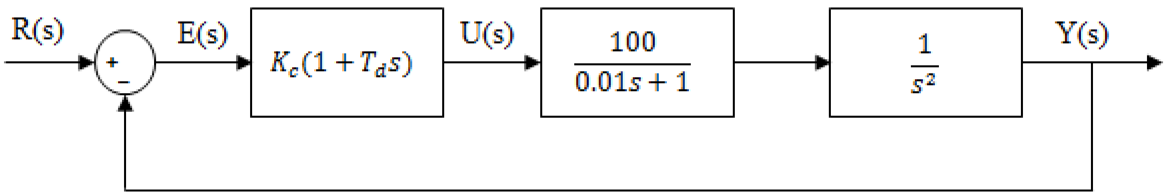


Figure 6.10: Closed Loop System with PD Controller in x-direction

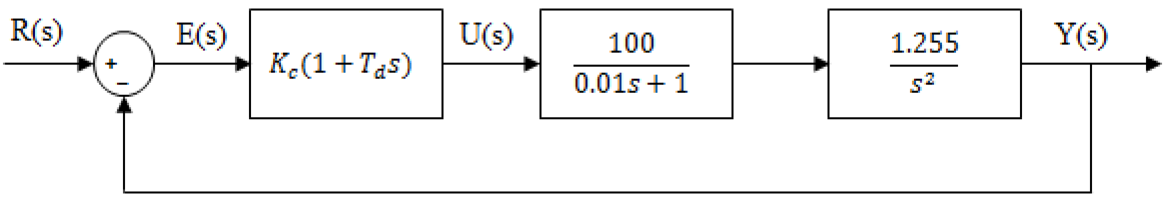


Figure 6.11: Closed Loop System with PD Controller in y-direction

6.5.1 Sample Design with PD controller

We will design a PD controller that satisfies 2% percentage overshoot and 1 second settling time with no steady state error in y-direction. Firstly, the closed loop transfer function and its characteristic polynomial are obtained in Figure 6.11:

$$T(s) = \frac{Y(s)}{R(s)} = \frac{F(s) * G_M(s) * G_{BBT,y}(s)}{1 + F(s) * G_M(s) * G_{BBT,y}(s)} = \frac{125K_c(T_d s + 1)}{0.01s^3 + s^2 + 125K_c(T_d s + 1)}$$

$$P_c(s) = 0.01s^3 + s^2 + 125K_cT_ds + 125K_c$$

We find the controller parameters with equal coefficients of the designed and desired characteristic polynomials, so the orders of them must be equal. If the order of the designed characteristic polynomial is higher than two, we add a multiplication described as the residue polynomial to the desired polynomial. For design of the PD controller, designed characteristic polynomial is third-order but we obtain a second-order desired characteristic polynomial from the performance measures. Therefore, the desired characteristic polynomial must contain a first order residue polynomial.

Secondly, ξ and ω_n are calculated from percentage overshoot (PO) and settling time (t_s):

$$\xi = \frac{-\ln(PO)}{\sqrt{\pi^2 + (\ln(PO)^2}} = \frac{-\ln(0.02)}{\sqrt{\pi^2 + (\ln(0.02)^2}} = 0.7797$$

$$\omega_n = \frac{4}{\xi * t_s} = \frac{4}{0.7797 * 1} = 5.13$$

Thirdly, residue polynomial is $(as + b)$, desired characteristic polynomial is determined:

$$P_D(s) = (as + b)(s^2 + 2\xi\omega_n s + \omega_n^2) = (as + b)(s^2 + 8s + 26.3185)$$

Finally, when desired and designed characteristic polynomials are equalized to each other, a, b, K_c and T_d are calculated.

$$a = 0.01 \quad b = 0.92 \quad K_c = 0.1937 \quad T_d = 0.315$$

As you can see on the graph, when the unit step input is applied to the system, percentage overshoot and settling time are almost same with our desired performance measures with no steady state error. PO equals to 0.266% and t_s equals to 0.906 second.

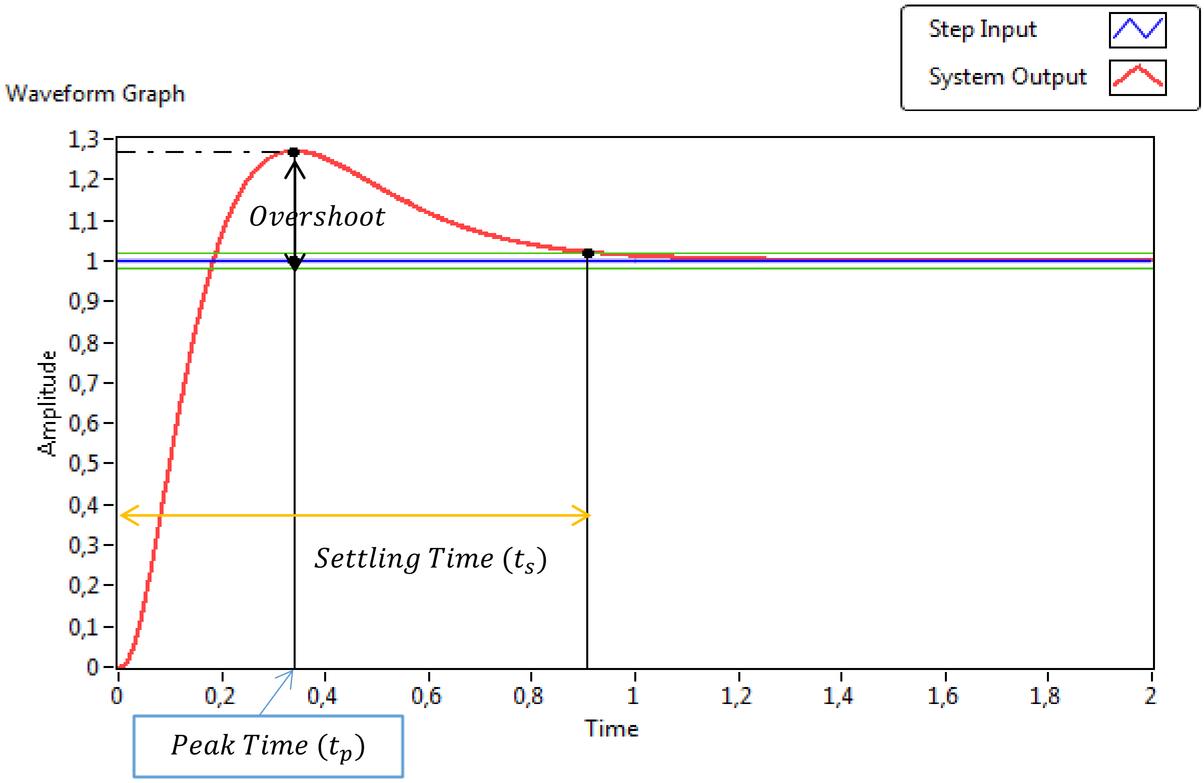


Figure 6.12: The Unit Step Response for Simulation with PD Controller on Y

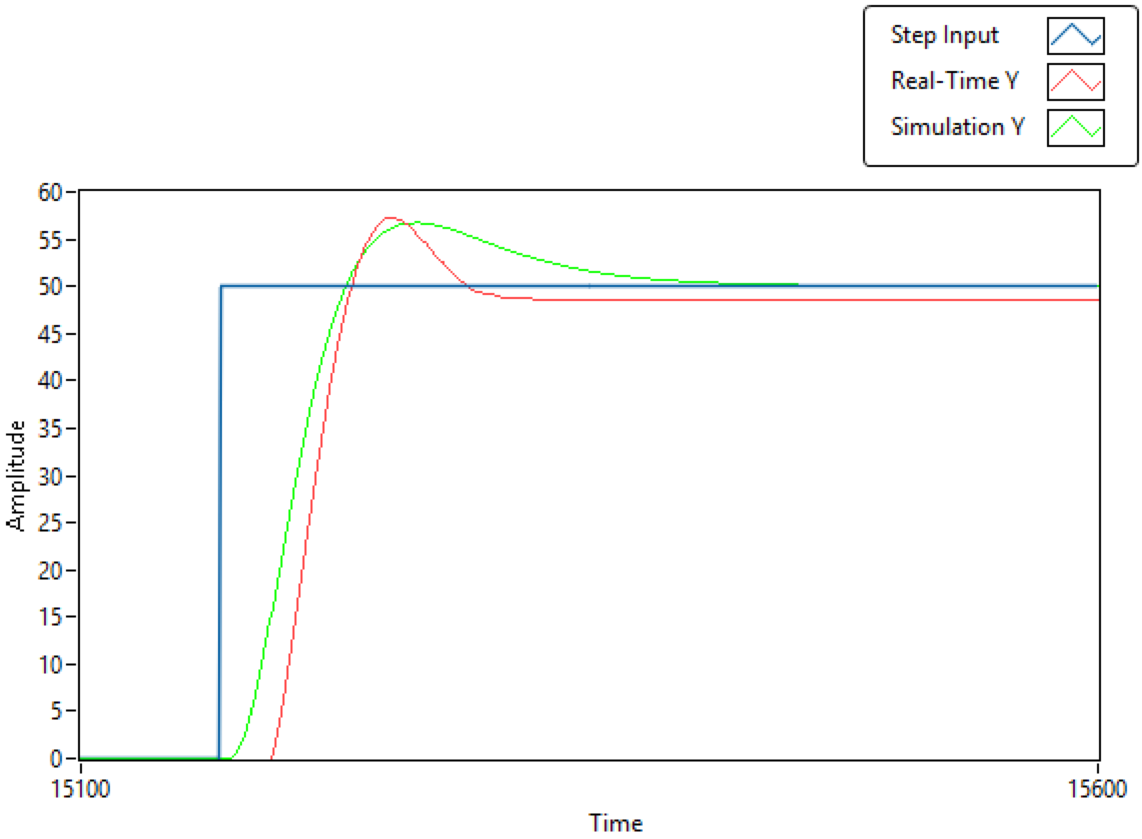


Figure 6.13: The Real-Time, Simulation and Step Input Graphs with PD Controller on Y

6.5.2 In-Lab Exercises

1. Open the “PDControl.slx”,
2. You will see the following block diagram shown below:

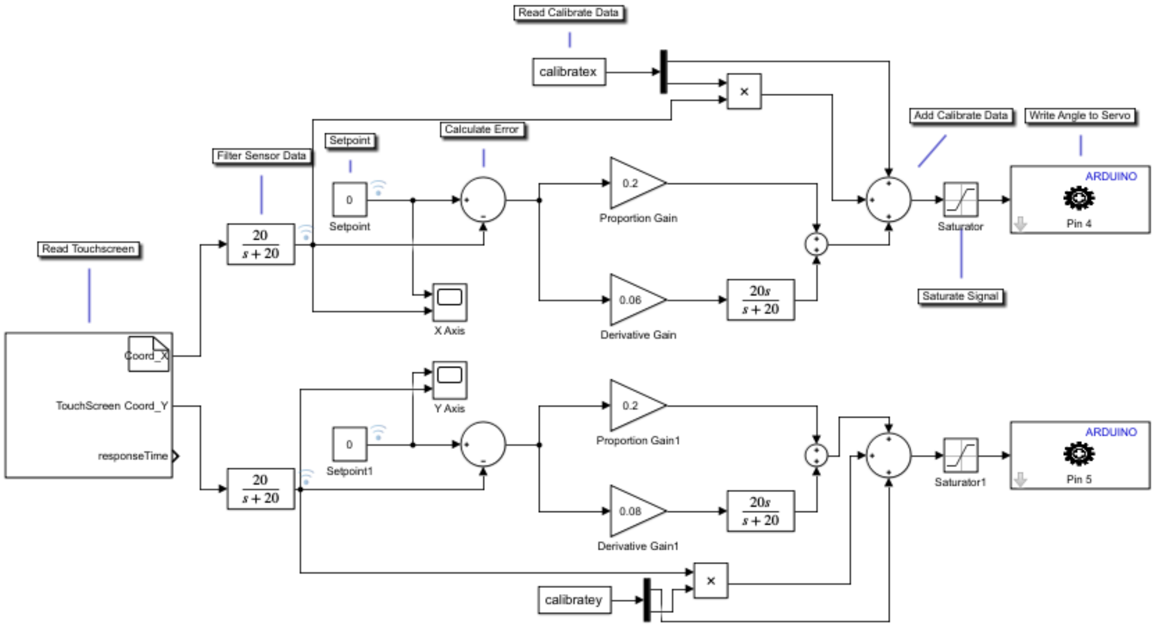


Figure 6.5.2: Block Diagram of “PDControl.slx”

3. Run the program. Double click to scope and open.
4. Observe real system responses.

Answer the following questions for the block diagram in x-direction as shown in Figure 6.10. Use the “BBT PD Controller Design” for this example.

1. Determine the order of the open loop transfer function.
2. Find the closed loop transfer function of the system with PD controller.
3. Obtain the characteristic polynomial of the closed loop system.
4. Determine the performance measures and calculate ξ and ω_n (Choose OS ≤ 5% and t_s ≤ 2 seconds).
5. Determine the residue polynomial and obtain the desired characteristic polynomial with the help of Step 4.
6. Equate the polynomials which are calculated at Step 3 and Step 5 and find the controller parameters.

7. What is the order of the system after controller? Does it change?

8. Draw the step response with designed PD controller. Is the closed loop system stable?

9. What is the value of steady state error?

10. Apply the designed controller parameters simulation and real time system. Discuss the differences between the response of simulation and real system. Are they consistent?

6.6 PV Controller

PV controller is one of the most popular servo position controllers. Contrary to PD controller, PV controller does not add a zero and a pole to the system. It only changes the root locus of the system. In other words, poles, which belong to the closed loop system, are moved to a more stable region on the root locus by PV controller. Moreover, PV controller prevents derivative kicks because the derivative term is on the feedback path. In addition, root locus is easily changed with a derivative term. Block diagrams are shown below:

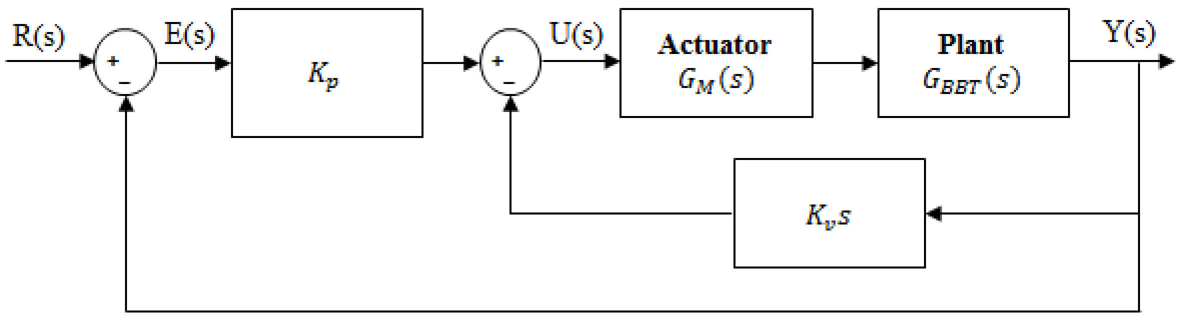


Figure 6.14: General Block Diagram of Closed Loop System with PV Controller

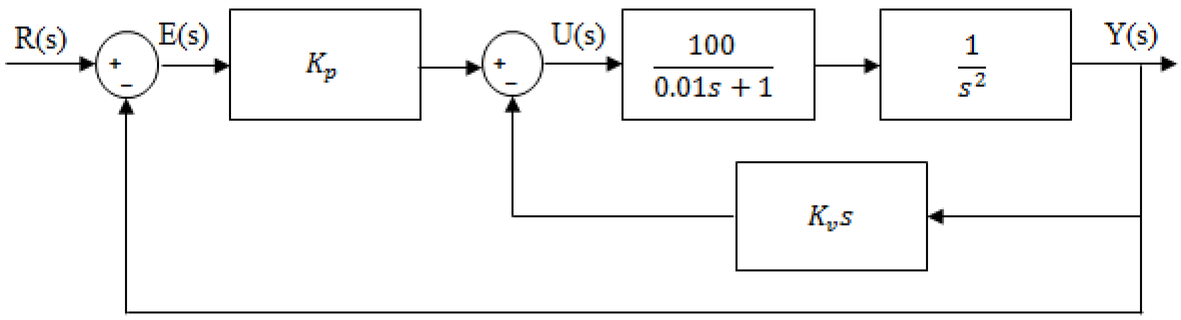


Figure 6.15: Closed Loop System with PV Controller in x-direction

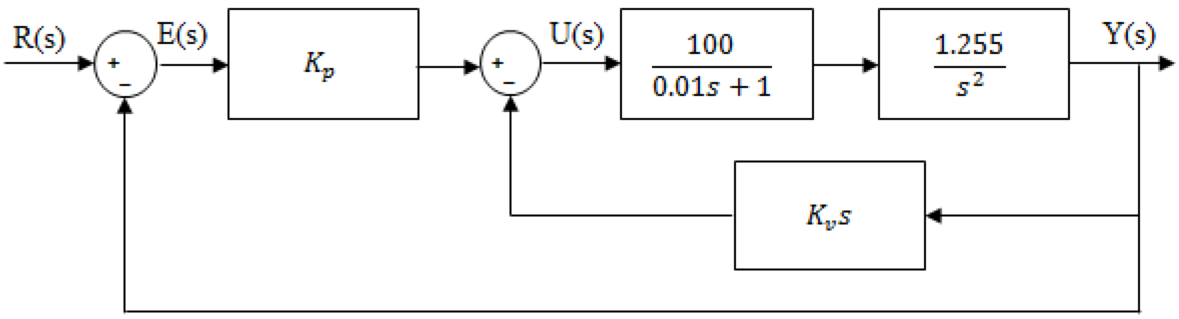


Figure 6.16: Closed Loop System with PV Controller in y-direction

6.6.1 Sample Design with PV controller

We will design a PV controller with respect to same performance measures at the PD controller. Gain part of the PV controller on the forward path represents $F_1(s)$ and derivative part of the PV controller on the feedback path represents $F_2(s)$. The inner loop transfer function is obtained and shown below:

$$T_{inner}(s) = \frac{G_M(s) * G_{BBT,y}(s)}{1 + F_2(s) * G_M(s) * G_{BBT,y}(s)} = \frac{125}{0.01s^3 + s^2 + 125K_v s}$$

Now, we obtain the outer loop transfer function using the inner loop transfer function.

$$T_{outer}(s) = \frac{T_{inner}(s) * F_1(s)}{1 + T_{inner}(s) * F_1(s)} = \frac{125K_p}{0.01s^3 + s^2 + 125K_v s + 125K_p}$$

PV controller is practically a PD controller because the characteristic polynomials of PV and PD controllers are the same. Parameters of the designed PD controller can be adapted to a PV controller easily. When our example is combined with example 6.5.1, PV parameters are calculated and shown below:

$$K_p = K_c = 0.1937 \qquad K_v = K_p * T_d = 0.061$$

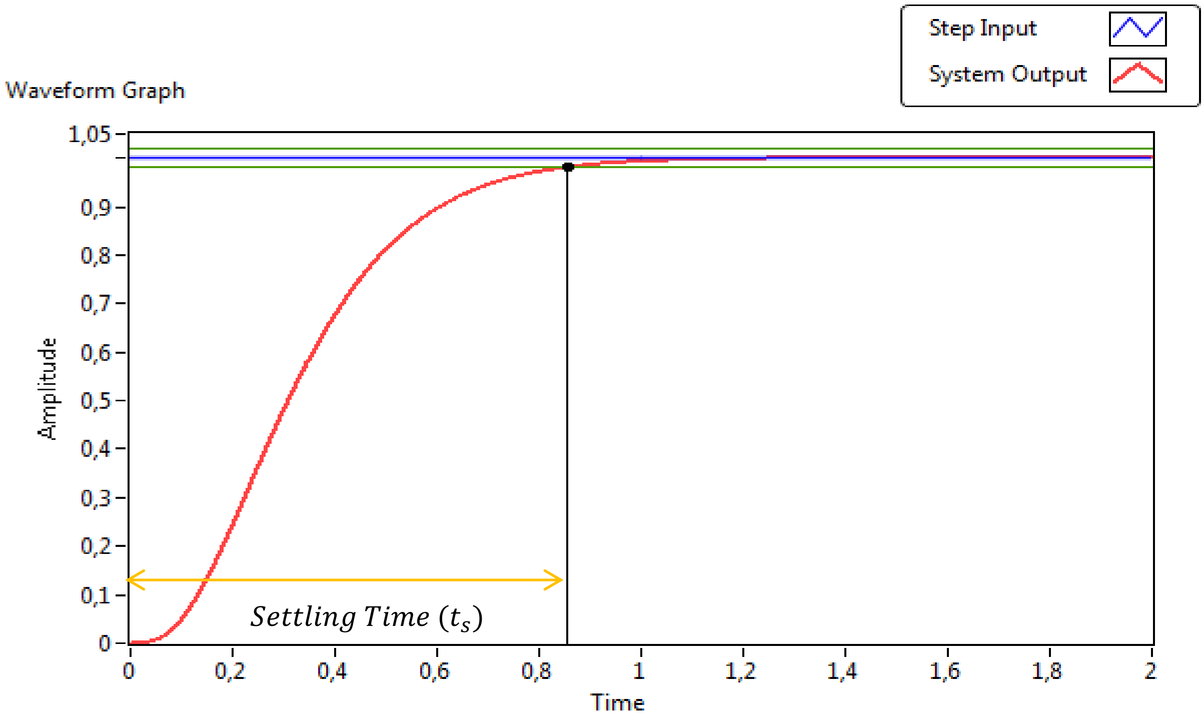


Figure 6.17: The Unit Step Response for Simulation with PV Controller on Y

In comparison with PD controller, PV controller decreases the settling time and eliminates the overshoot. The closed loop system has also no steady state error. The settling time of the closed loop system is 0.853 second.

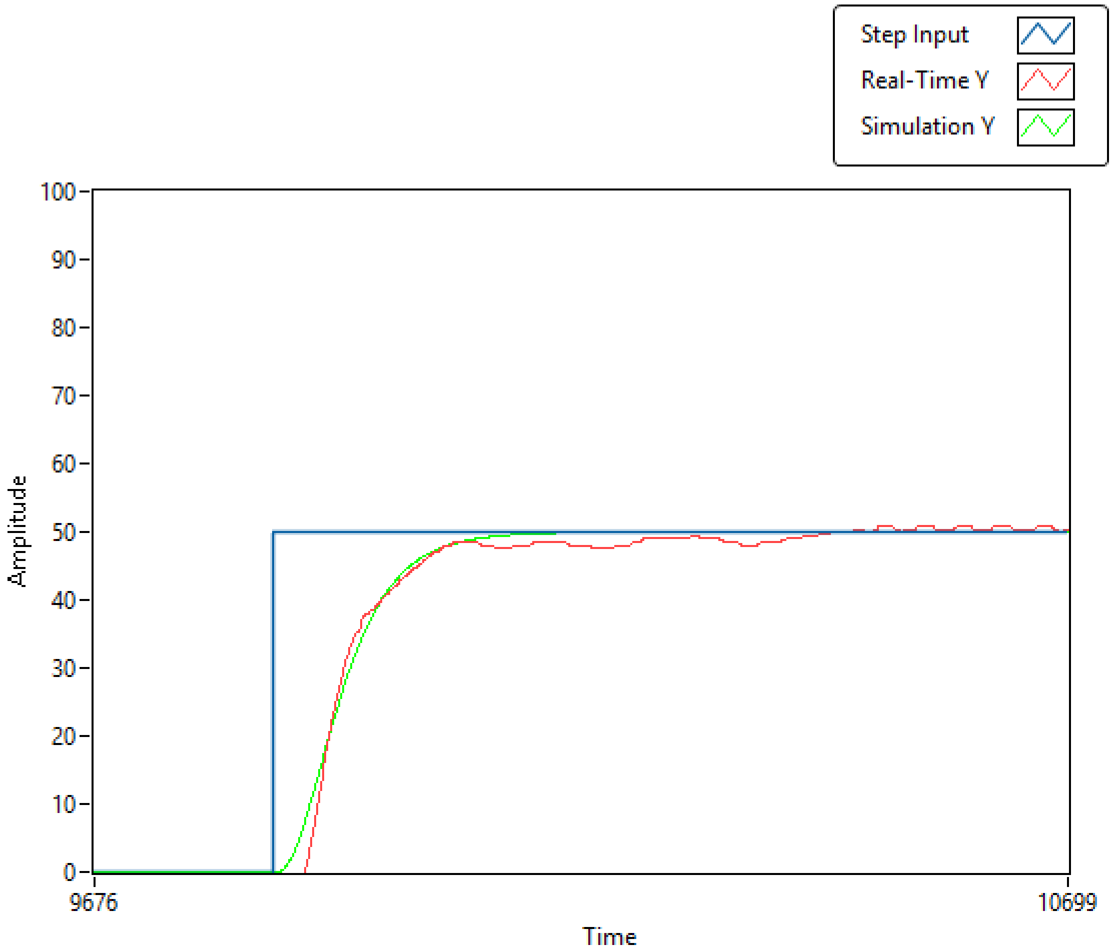


Figure 6.18: The Real-Time, Simulation and Step Input Graphs with PV Controller on Y

6.6.2 In-Lab Exercises

1. Open the "PVControl.slx",
2. You will see the following block diagram shown below:

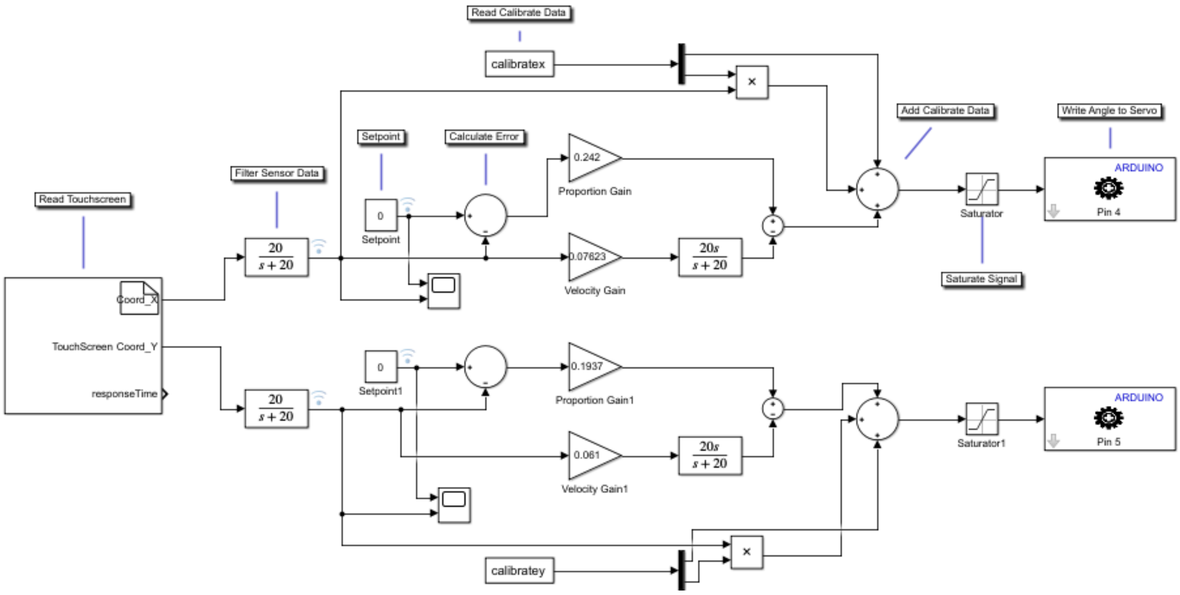


Figure 6.6.2: Block Diagram of "PVControl.slx"

3. Run the program. Double click to scope and open.
4. Observe real system responses

Answer the following questions for the block diagram as shown in Figure 6.15. Use the "BBT PV Controller Design" for this example.

1. Determine the order of the open loop transfer function.
2. Find the closed loop transfer function of the system with PV controller.
3. Obtain the characteristic polynomial of the closed loop system.
4. Determine the performance measures and calculate ξ and ω_n (Choose OS ≤ 5% and t_s ≤ 2 seconds).
5. Determine the residue polynomial and obtain the desired characteristic polynomial with the help of Step 4.
6. Equate the polynomials which are calculated at Step 3 and Step 5 and find the controller parameters.
7. What is the order of the system after adding the controller? Did it change?
8. Draw the step response with designed PV controller. Is the closed loop system stable?
9. With respect to PD controller, what does it change?

10. What is the value of steady state error?

11. Apply the designed controller parameters simulation and real time system. Discuss the differences between the response of the simulation and the real system. Are they consistent?

6.7 PID Controller

PID controller is the most popular controller when using non-linear applications. The types of non-linear controllers are complicated and hardly implemented, so the PID controller is preferred even if the system is linear. The controller adds two zeros and one pole to the system. Mainly, the controller places the closed loop poles to a desired location. Nevertheless, like PI controller, PID controller does not usually return an accurate result for the position control. The closed loop poles can be stable in a small region within plant's integrator poles. Block diagrams with the PID controller are shown below:

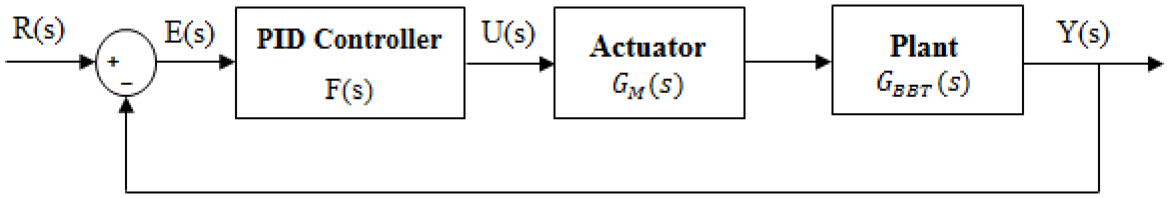


Figure 6.19: Closed Loop System with PID Controller

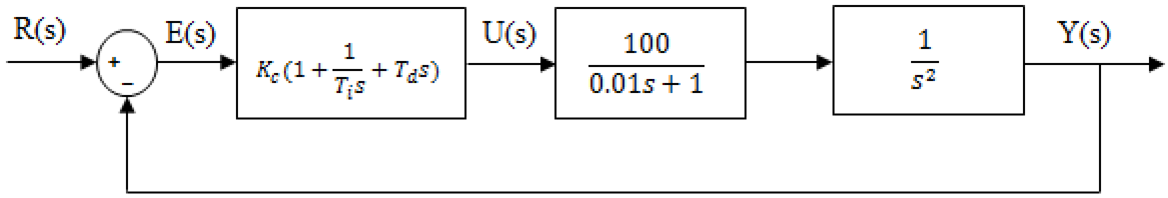


Figure 6.20: Closed Loop System with Academic PID Controller in x-direction

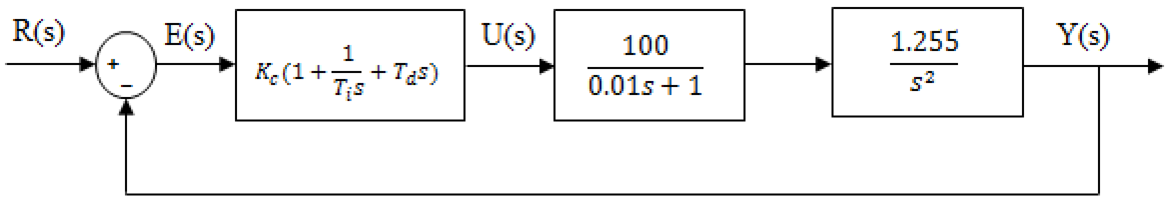


Figure 6.21: Closed Loop System with Academic PID Controller in y-direction

When parallel PID parameters are increased independently, Table 6.1 shows the qualification results of the performance measures.

Table 6.1: Qualification Results Corresponding Increase in PID Parameters

PID Parameters	Peak 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	No effect

6.7.1 Sample Design with PID controller

A PID controller will be designed that satisfies 5% overshoot and 3 seconds settling time with no steady state error in y-direction. Thanks to integrator, there is no steady state error. Firstly, closed loop transfer function and its characteristic polynomial are obtained in Figure 6.21:

$$T(s) = \frac{Y(s)}{R(s)} = \frac{F(s) * G_M(s) * G_{BBT,y}(s)}{1 + F(s) * G_M(s) * G_{BBT,y}(s)}$$

$$= \frac{125K_c(T_iT_ds^2 + T_is + 1)}{0.01T_is^4 + T_is^3 + 125K_c(T_iT_ds^2 + T_is + 1)}$$

$$P_c(s) = 0.01T_is^4 + T_is^3 + 125K_cT_iT_ds^2 + 125K_cT_is + 125K_c$$

Secondly, ξ and ω_n are calculated using percentage overshoot (PO) and settling time (t_s):

$$\xi = \frac{-\ln(PO)}{\sqrt{\pi^2 + (\ln(PO)^2}} = \frac{-\ln(0.05)}{\sqrt{\pi^2 + (\ln(0.05)^2}} = 0.69$$

$$\omega_n = \frac{4}{\xi * t_s} = \frac{4}{0.7797 * 3} = 1.932$$

Thirdly, residue polynomial is $(as^2 + bs + c)$, desired characteristic polynomial is determined:

$$P_D(s) = (as^2 + bs + c)(s^2 + 2\xi\omega_ns + \omega_n^2) = (as^2 + bs + c)(s^2 + 8s + 26.3185)$$

Finally, a, b, c, K_c, T_i and T_d are found by solution designed and desired characteristic polynomials:

$$a = 0.00866 \quad b = 0.8433 \quad c = 5.5463 \quad K_c = 0.1656 \quad T_i = 0.8664 \quad T_d = 0.4364$$

Controller parameters are written for the closed loop system; simulation step response is shown in Figure 6.22.

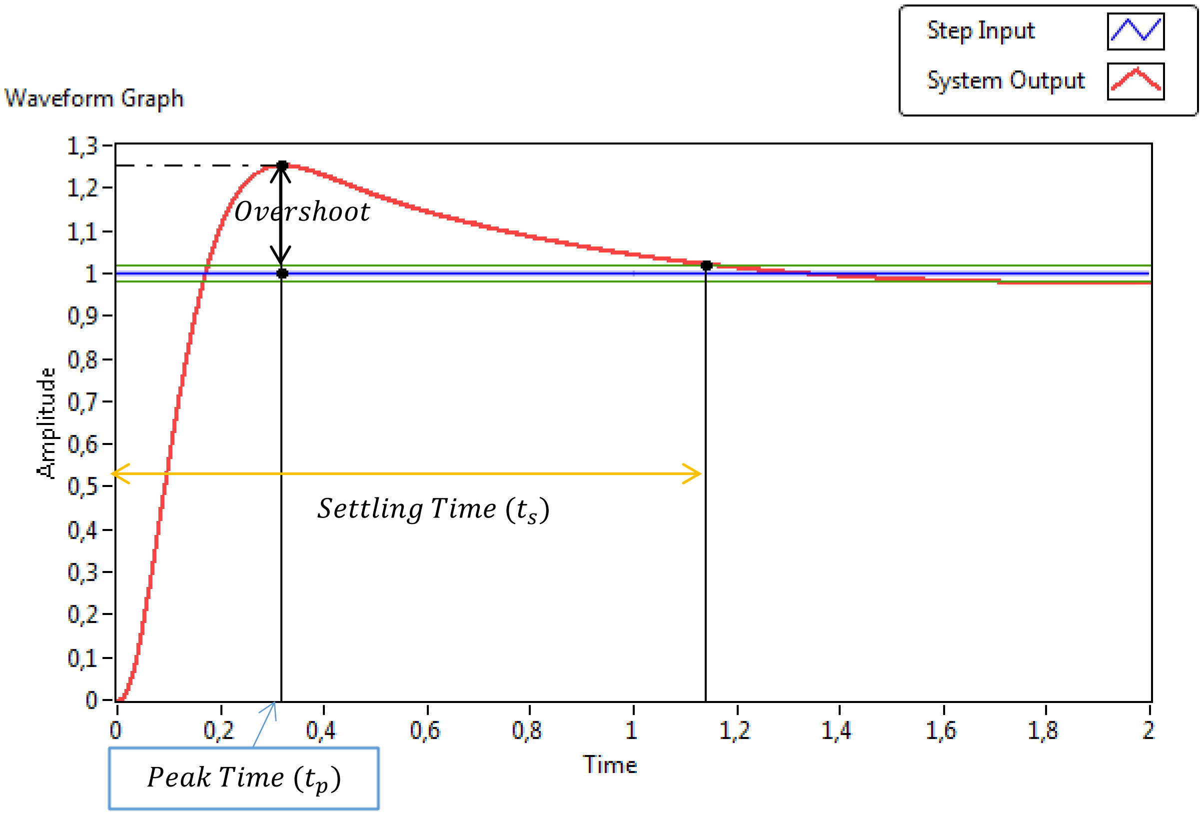


Figure 6.22: The Unit Step Response for Simulation with PID controller on Y

Overshoot is 25.147% and settling time is 1.14 seconds. Closed loop system does not satisfy the expected overshoot. PID controller design is not preferred for ball balancing table because of the system's integrator terms. In comparison to PV and PD, overshoot is very high. Also, the system is not as fast as PV and PD.

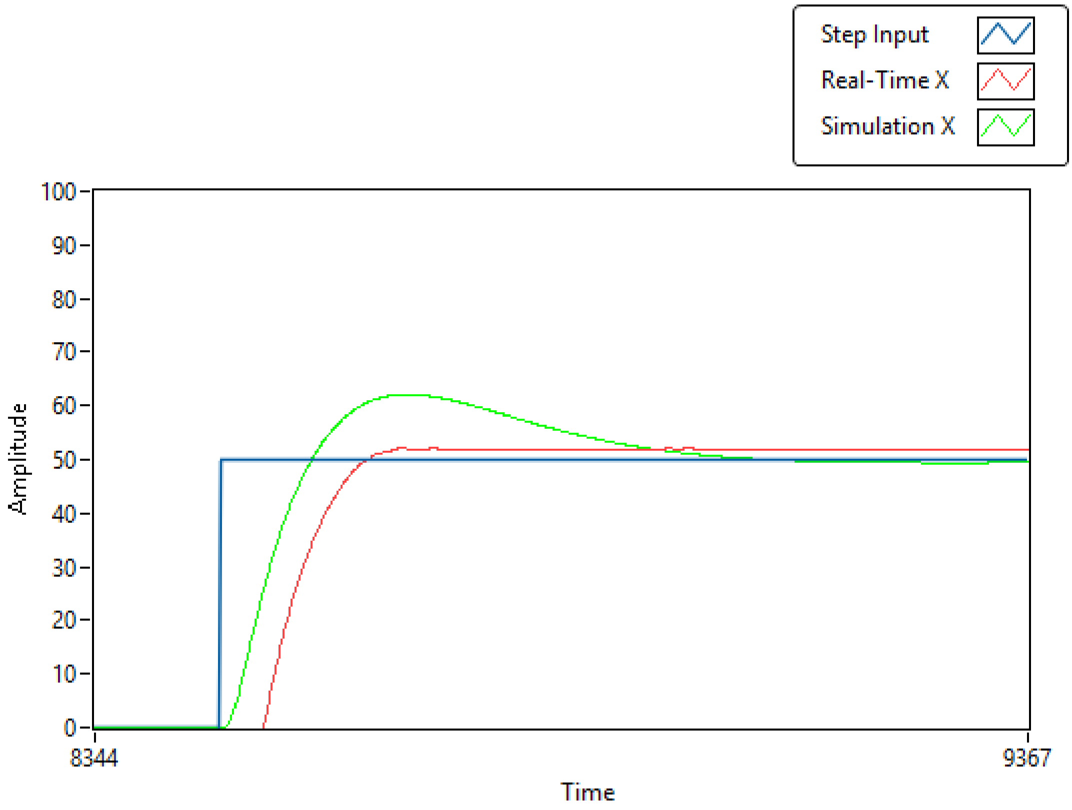


Figure 6.23: The Real-Time, Simulation and Step Input Graphs with PID Controller on Y

6.7.2 In-Lab Exercises

1. Open the “PIDControl.slx”,
2. You will see the following block diagram shown below:

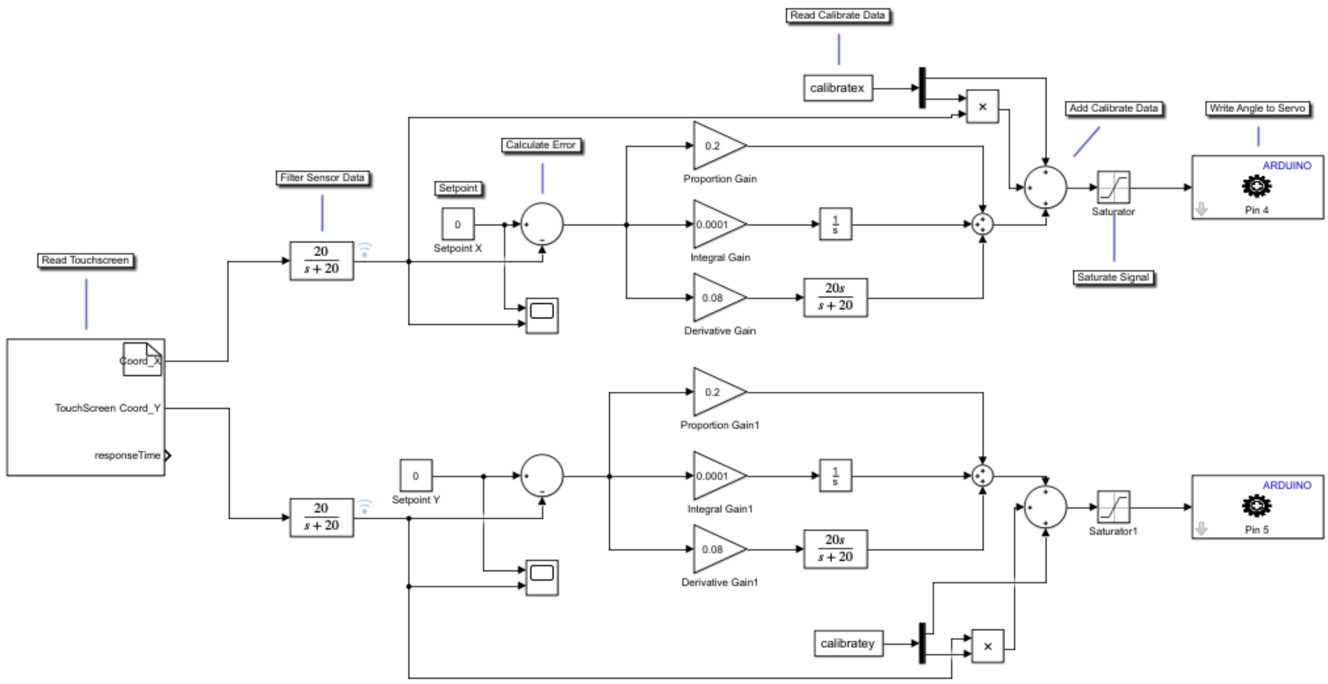


Figure 6.7.2: Block Diagram of “PIDControl.slx”

3. Run the program. Double click to scope and open.
4. Observe real system responses.

Answer the following questions for the block diagram in x-direction as shown Figure 6.20. Use the “BBT PID Controller Design” for this example.

1. Determine the order of the open loop transfer function.
2. Find the closed loop transfer function of the system with PID controller.
3. Obtain the characteristic polynomial of the closed loop system.
4. Determine the performance measures and calculate ξ and ω_n (Choose OS ≤ 10% and $t_s \leq 5$ seconds).
5. Determine the residue polynomial and obtain the desired characteristic polynomial with the help of Step 4.
6. Equate the polynomials which are calculated at Step 3 and Step 5 and find controller parameters.
7. What is the order of the system after controller? Does it change?
8. Draw the step response with designed PID controller. Is the closed loop system stable?
9. What is the value of steady state error?
10. Apply the designed controller parameters to simulation and real time system. Discuss the differences between the responses of the simulation and the real system. Are they consistent?

7 Frequency Response Analysis

7.1 Introduction

For dynamic systems, two main responses can be determined; Time Response and Frequency Response. System responses also can be divided into two times intervals; Transient Response and Steady State Response. In time response analysis, transient response characteristics are the significant part of the response. Transient response of a system is determined for step and ramp inputs.

In frequency response analysis; the input is a sinusoidal function and the significant part of the response is steady state response. For linear time-invariant systems, response of a sinusoidal input is also a sinusoidal response with the same frequency.

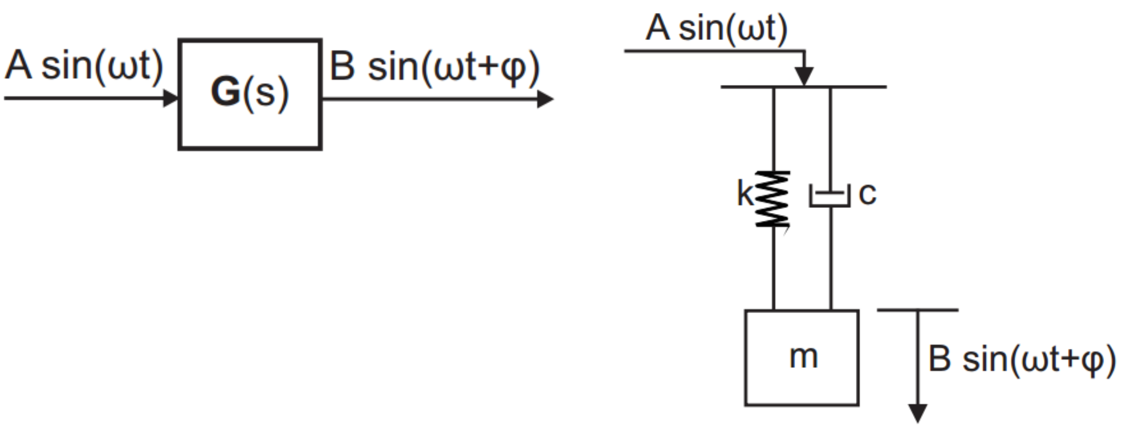


Figure 7.1: Sinusoidal Input and System Response

For mass-damper-spring system shown in Figure 7.1, if a sinusoidal input is applied from the top edge, mass block moves as a sinusoidal form with the same frequency. But, in general, amplitude of motions is not equal (A≠B) and a phase difference occurs (ϕ≠0).

In Figure 2, relationship between input and output can be seen clearly. The response of the input signal is scaled and delayed. The scale (amplitude ratio) and delay (phase) are functions of the input frequency.

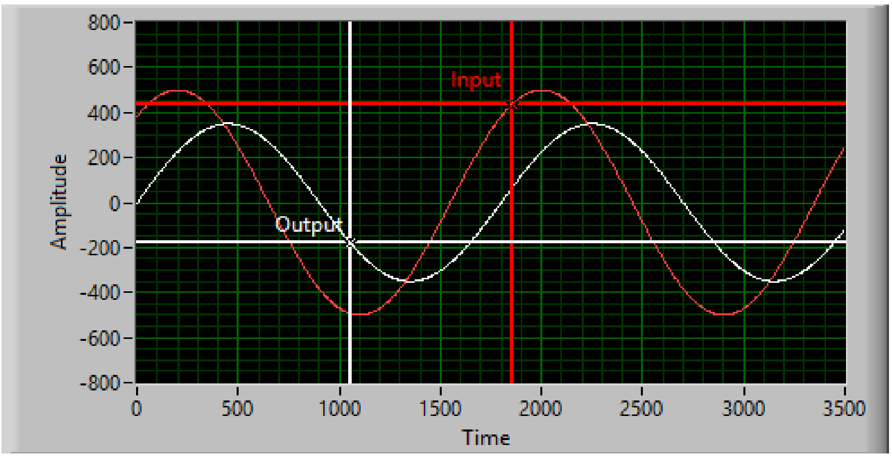


Figure 7.2: Sinusoidal Input and Output Relationship

These Input – Output Relationship can be expressed by **Bode Diagrams**. Bode Diagrams involve two separate diagrams; Amplitude Ratio vs. Frequency (Logarithmic) and Phase vs. Frequency.

7.2 Frequency Response of a Linear System

The amplitude ratio and the phase relationships are obtained from the transfer function of the system. To obtain this, simply s=jω transformation should be applied, so G(jω) is obtained.

The amplitude ratio of the system equals to the module of the system and phase of the system equals to the argument of the system.

$$Amplitude: |G(j\omega)| = \frac{B}{A}, \qquad Phase: \angle G(j\omega) = \varphi$$

For an imaginary number Z = a + jb;

$$|Z| = \sqrt{a^2 + b^2}, \qquad \angle Z = \tan^{-1}\left(\frac{B}{A}\right)$$

Now let's apply this to a simple transfer function;

$$G(s) = \frac{10(s+2)}{s(5s+3)} = 10(s+2)\frac{1}{s}\frac{1}{(5s+3)}$$

So, amplitude ratio;

$$|G(j\omega)| = \left|10(j\omega+2)\frac{1}{j\omega}\frac{1}{5j\omega+3}\right| = \frac{|10||j\omega+2|}{|j\omega||5j\omega+3|} = \frac{10\sqrt{\omega^2+4}}{\omega\sqrt{25\omega^2+9}}$$

And the phase angle;

$$\angle G(j\omega) = \angle 10 + \angle(j\omega + 2) - \angle(j\omega) - \angle(5j\omega + 3) = 0 + \tan^{-1}\frac{\omega}{2} - \frac{\pi}{2} - \tan^{-1}\frac{5\omega}{3}$$

7.3 In Lab Exercise: Frequency Response of BBT

As mentioned before, the Bode Diagram includes two separate diagrams such as Phase – Frequency and Amplitude Ratio – Frequency. In this lab exercise, the relationship between frequency and amplitude will be examined. Amplitude Ratio – Frequency diagram of Ball Balancing Table control system will be shown experimentally.

The amplitude-frequency relationship diagram obtained in the simulation is shown in Figure 7.3.

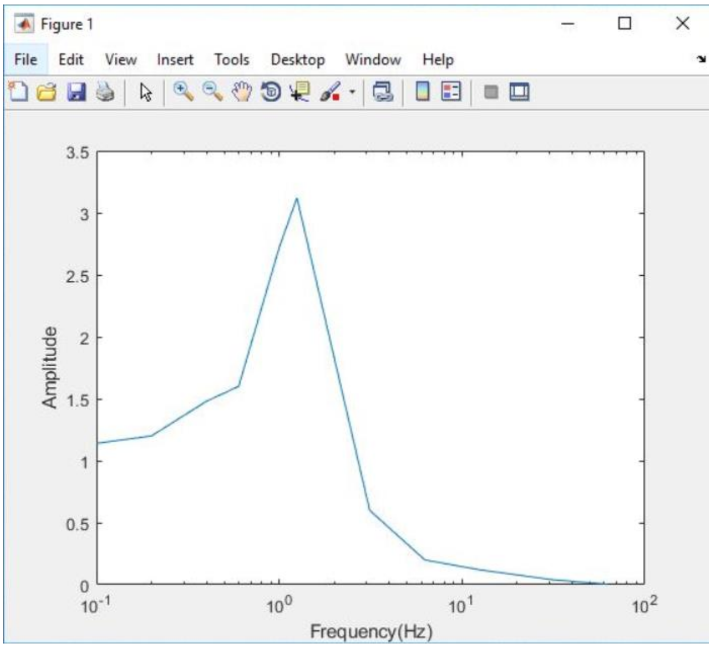


Figure 7.3: Magnitude Plot

Follow the instructions below to experience the frequency response of the Ball Balancing Table control system

1. Open the “FrequencyResponse.slx”,
2. You will see the following block diagram shown below:

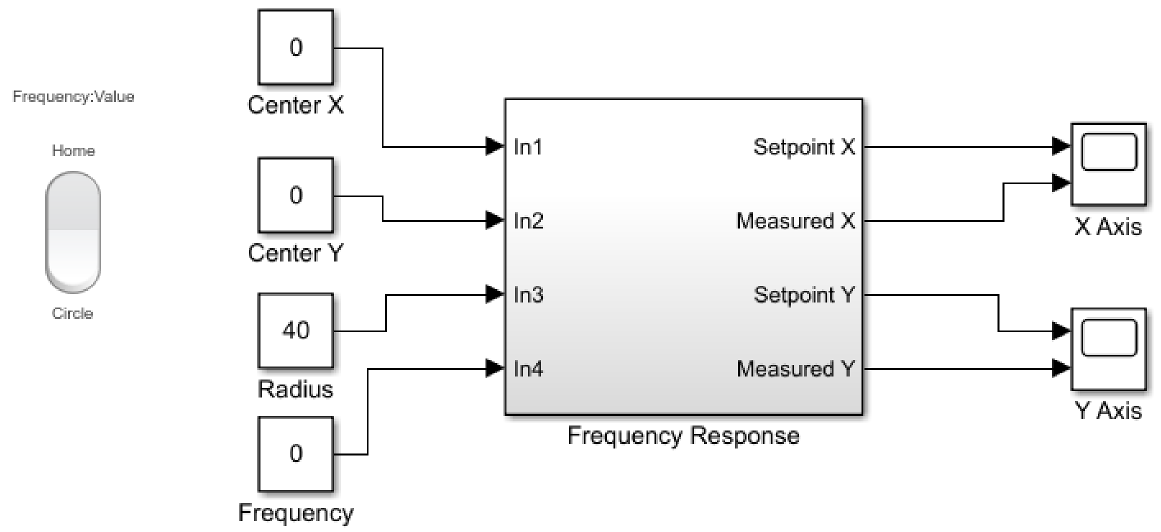


Figure 7.4: Block Diagram of FrequencyResponse.slx

3. Run the program. Double click to scopes and open.
4. Change Center X, Center Y, Radius(using switch) and Frequency values and observe real system responses.

The yellow data in the graph is the real-time data of the ball measured from the system. The data obtained on the X axis at a frequency of 10 Hz is shown in Figure 7.5.

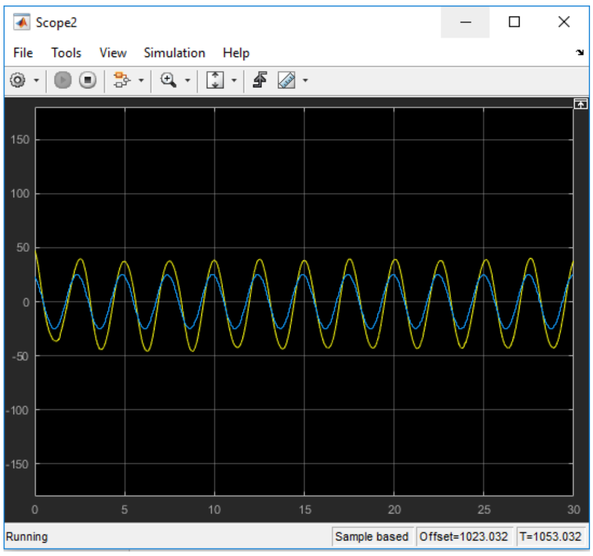


Figure 7.5: X axis data at 10 Hz Frequency

The data from the X axis at a frequency of 20 Hz is shown in Figure 7.6.

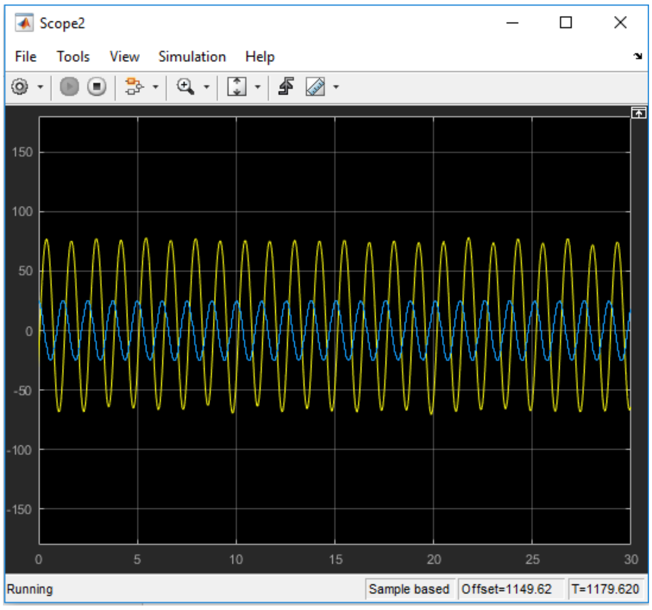


Figure 7.6: X axis data at 20 Hz Frequency

The data from the X axis at a frequency of 100 Hz is shown in Figure 7.7.

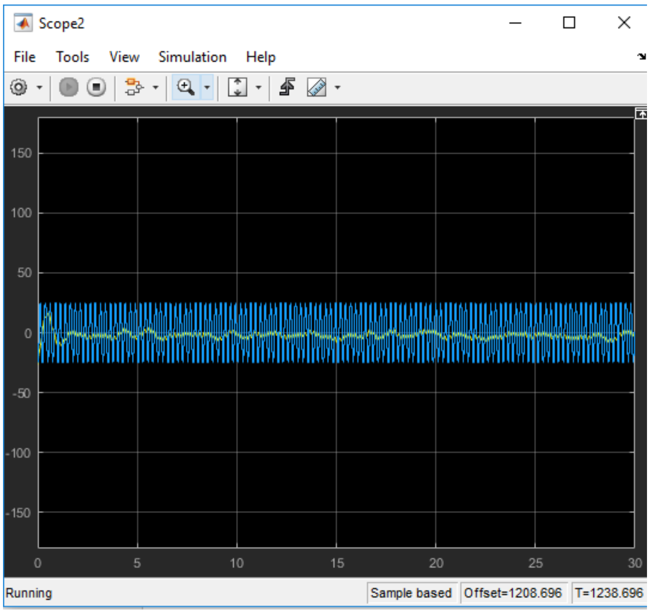


Figure 7.7: X axis data at 100 Hz Frequency

The data from the X axis at a frequency of 1000 Hz is shown in Figure 7.8.

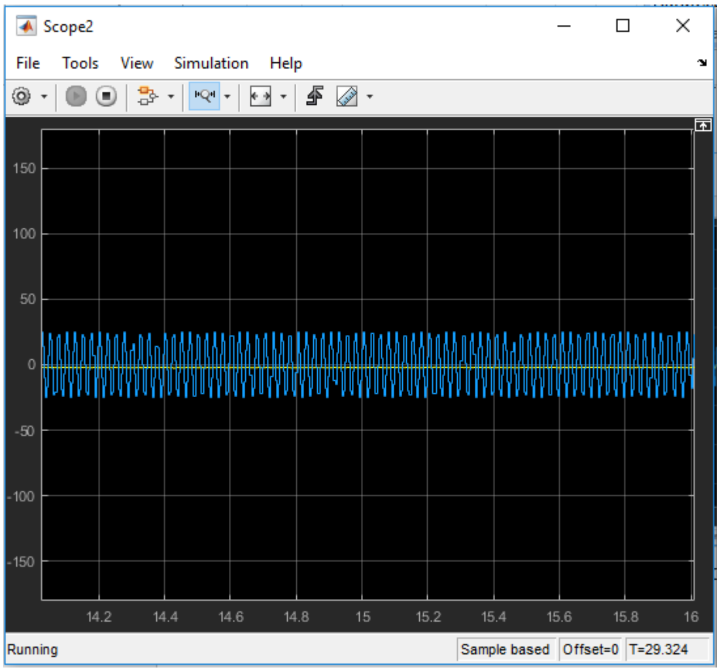


Figure 7.8: X axis data at 1000 Hz Frequency

5. Please turn the switch to "HOME" position before stopping the program.

References

1. M. M. Wu Zhi Qiao, «PID type fuzzy controller and parameters adaptive method, » *Fuzzy Sets and Systems,* no. 78, pp. 23-35, 1996.
2. S. Y. Kevin M. Passino, Fuzzy Control, An Imprint of Addison Wesley Longman, Inc., 1997.