

©2015 Acrome Inc., All rights reserved.

Acrome Inc.
ITU ARI4 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 Acrome myControl Team for their supports in embedded control and preparation of the courseware.

Contents

1	Components of Ball and Beam.....	4
1.1	RC Servo Motors.....	4
1.1.1	Overview	4
1.1.2	Theory of Operation	6
1.2	Potentiometer Sensor	7
1.3	NI myRIO	8
1.4	Power Supply Box.....	9
1.5	Mechanics of Ball and Beam.....	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 a RC Servomotor with PWM	11
3	Filtering	13
3.1	In-Lab Exercise.....	13
4	System Modelling.....	15
4.1	Non-Linear Equation of Motion	16
4.1.1	Lagrangian Method	16
4.1.2	Newton's Law of Motion	18
4.2	Modelling of Ball and Beam System.....	19
4.2.1	Linearization around Operating Point.....	20
4.2.2	Obtaining Transfer Function of Plant.....	21
4.3	Modelling of Actuator	21
4.4	Cascade Control of Ball and Beam	22
5	System Identification.....	24
5.1	Collecting Estimation Data.....	24
5.1.1	In Lab Exercise:.....	24
5.2	Obtaining Estimated Model via System Identification.....	25
5.2.1	In Lab Exercise:.....	25
6	Performance Measures	27
6.1	Understanding Percentage Overshoot, Peak Time, Settling Time and Steady State Error	27
6.1.1	Damping Ratio (ξ).....	27
6.1.2	Natural Frequency (ω_n).....	27

6.1.3	Percentage Overshoot.....	27
6.1.4	Peak Time (t_p).....	29
6.1.5	Settling Time (t_s).....	29
6.1.6	Steady State Error	29
6.1.7	In-Lab Exercise.....	30
7	Control System Design.....	31
7.1	Academic PID controller	31
7.2	Parallel PID controller	32
7.3	Root Locus.....	32
7.4	P Controller	33
7.4.1	Sample Design with P Controller.....	33
7.4.2	In-Lab Exercise.....	35
7.5	PD Controller	37
7.5.1	Sample Design with PD controller.....	37
7.5.2	In-Lab Exercises	39
7.6	PV Controller	40
7.6.1	Sample Design with PV controller	41
7.6.2	In-Lab Exercises	42
7.7	PID Controller.....	43
7.7.1	Sample Design with PID controller.....	44
7.7.2	In-Lab Exercises	45
7.8	Fuzzy Logic Controller.....	46
7.8.1	Fuzzy Sets.....	47
7.8.2	Membership Function	47
7.8.3	Operations with Fuzzy Sets.....	47
7.8.4	Fuzzy Control Structure	48
7.8.5	Fuzzy Rule Structures	49
7.8.6	Design of the Fuzzy Logic Controller	50
	References	57

1 Components of Ball and Beam

All the main parts of “ACROME Ball and Beam” can be seen below.

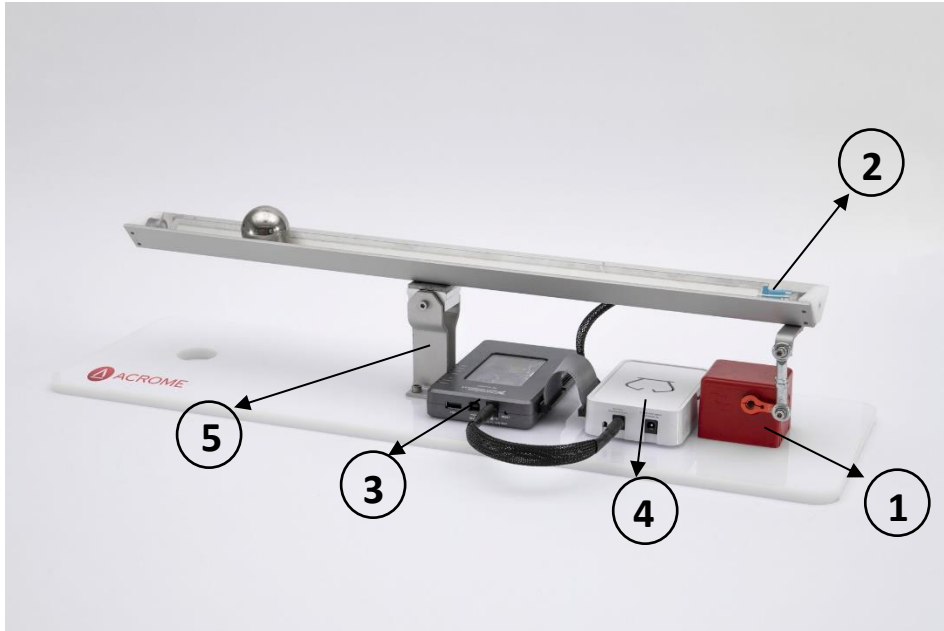


Figure 1.1: Components: 1. RC Servo 2. Potentiometer Sensor 3. myRIO 4. Power Supply Box
5. Mechanics of Beam

1.1 RC Servo Motors

1.1.1 Overview

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



Figure 1.2: RC Servo Motor

RC servo motor consists of several parts which include:

Controller: Circuit that is responsible of reading the control signals (PWM signals –more about PWM signal can be found in Chapter 2) and controls the motor accordingly. Controller circuits determine the type of servos; they can be either digital or analog. Analog servos handle PWM signal 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 have much better performance 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 drive shaft so the rotation of the drive shaft results in different resistances on the potentiometer. By reading the resistance values, controller knows exactly at which angle the driveshaft is.

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

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

Drive shaft: 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 carries “+”, “-” and “signal” to the controller. Connectors can have different color codes depending on the manufacturer.

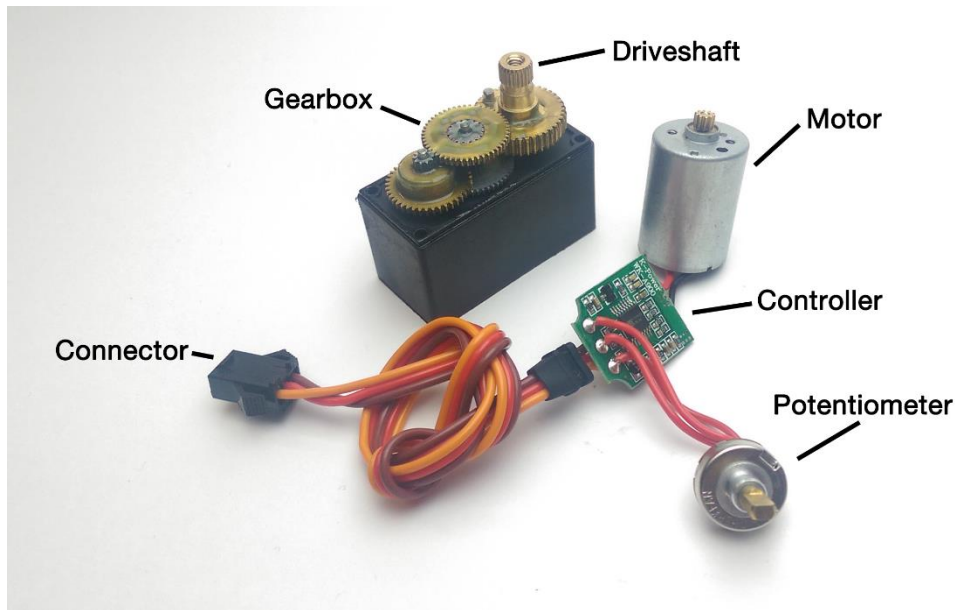


Figure 1.3: RC Servo Components

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 to 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 to the H-bridge output.

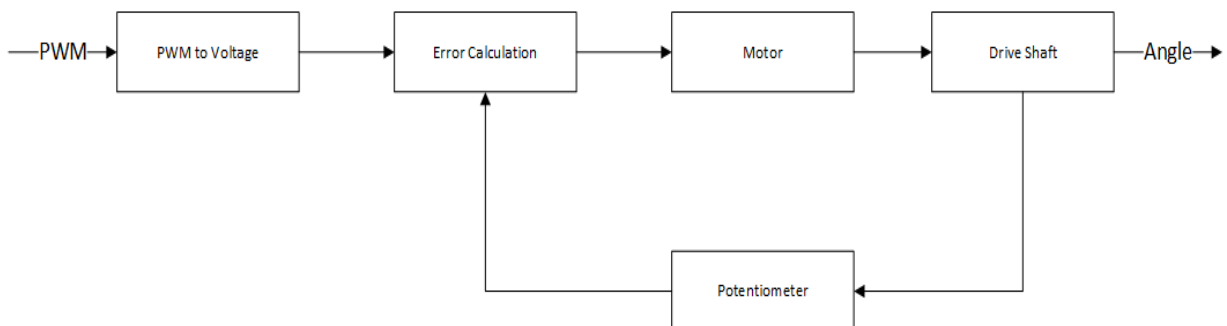


Figure 1.4: RC Servo Theory of Operation

1.2 Potentiometer Sensor

Position feedback of the ball is acquired from potentiometer sensor which is shown in below.



Figure 1.5: Potentiometer Sensor

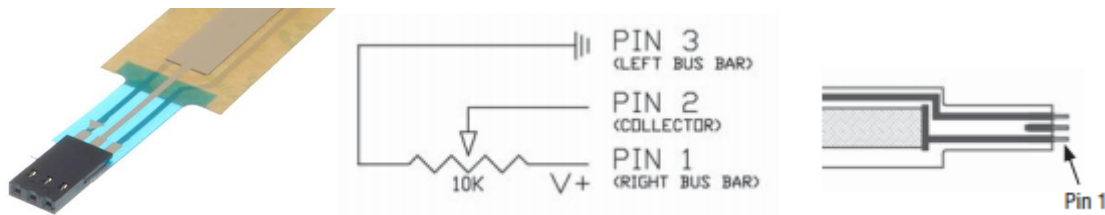


Figure 1.6: Pins of the Potentiometer Sensor

Potentiometer sensor's length is 50 cm. As you can see in Figure 1.6, there are 3 pins of the potentiometer that carries "+", "-" and "collector". Pin 3 is connected to the ground as "-" terminal, pin 1 is connected to 5 volt as "+" terminal and pin 2 is connected to the analog input.

Sensor working principle can be explained in this way; ball presses down on various part of the strip, the resistance linearly changes from 100Ω to 10.000Ω . Relative position of the ball is read from via collector (pin 2) of the sensor as analog voltage the range of 0 to 5 volt. Pin 2 is connected analog input of myRIO. myRIO is converted the position data analog input to digital input with high resolution. In other words, DAC (Digital Analog Converter) of myRIO is proportioned the position 5 to 1024. As a result resolution of the position data is:

$$Resolution = \frac{5 - 0}{2^{10}} \cong 0.00488$$

1.3 NI myRIO

myRIO is portable embedded device that students and hobbyist can design control, robotic and mechatronic systems.



Figure 1.7: NI myRIO

myRIO has many ports and sensors on board including; analog inputs (AI), analog outputs (AO), digital inputs and outputs (DIO), audio and power outputs, an accelerometer and LED's. Also, myRIO has a full size USB port that can be utilized as host and wireless 802.11.bgn.

myRIO differs from many other embedded controllers with its on board FPGA (Field Programmable Gate Array) chip. FPGA enables blazing fast I/O operations (as low as 0.25 nanosecond cycle time). FPGA is working together with a 667 MHz 2 core application processor with LabVIEW Real-time OS running on it.

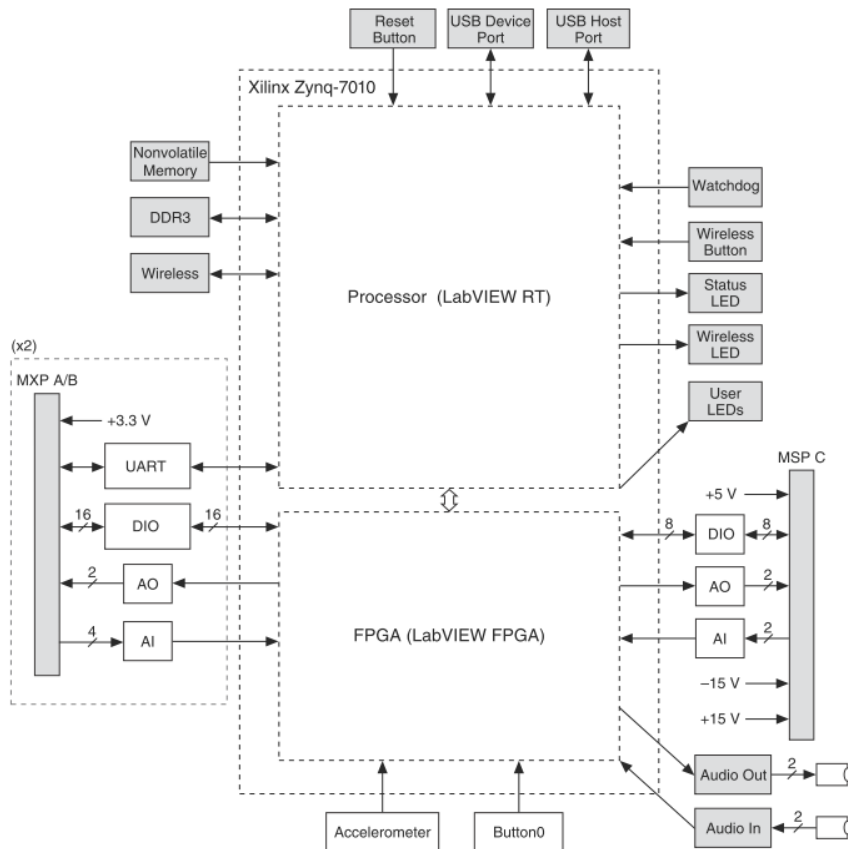


Figure 1.8: myRIO Ports and Peripherals

1.4 Power Supply Box

ACROME Ball and Beam Power Supply Box is responsible for delivering the necessary power to the system, with its 5V power supply inside. myRIO and potentiometer sensor is feed from myRIO power source. Beside power delivery, box serves as an interconnector between the beam and myRIO by keeping all the necessary wiring under the hood.

1.5 Mechanics of Ball and Beam

Beam is connected to the base plate of the system over a cord joint with one degree of freedom. Servo motor is connected to the beam over link with rod and ball joint. Potentiometer sensor is stuck on the beam.

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 the 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:

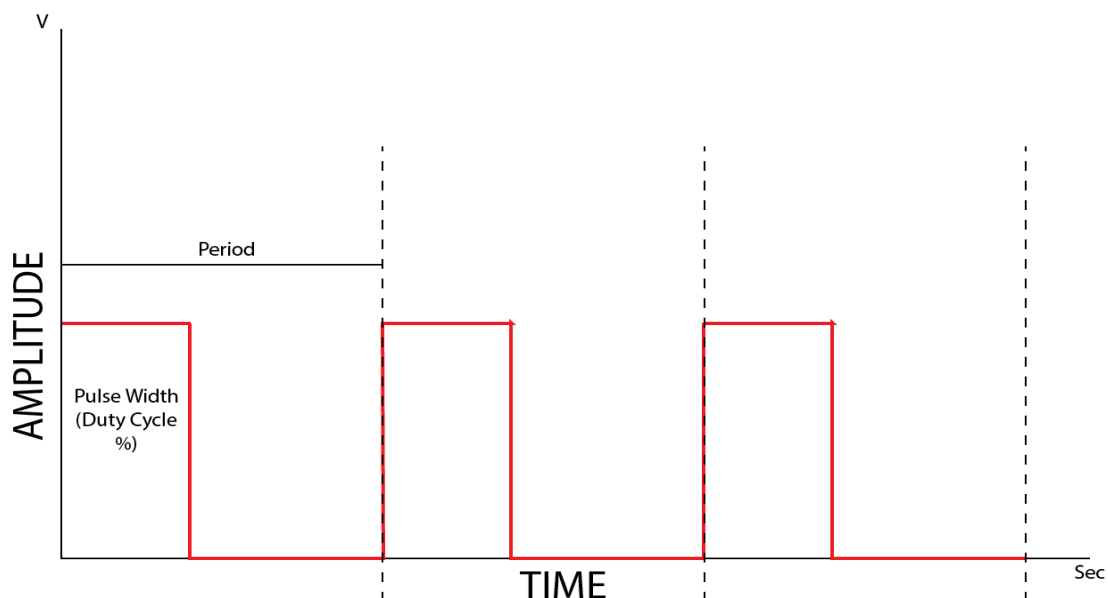


Figure 2.1: Typical PWM signal with its properties

2.1.1 In-Lab Exercise

1. Open “Generating and Reading PWM.vi” where Figure 2.2 shows its front panel.

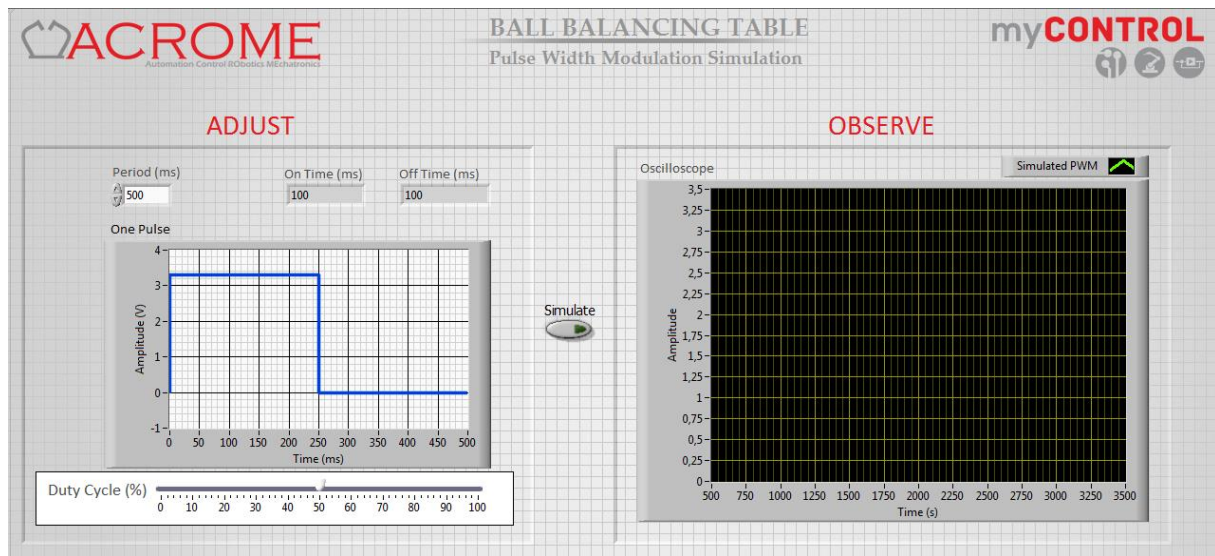


Figure 2.2: Front panel of Generating and Reading PWM.vi

2. Change percentage duty cycle and the period as desired,
3. Run the VI,
4. Observe simulated PWM from oscilloscope,
5. Compare the consistency of the performance obtained from ‘Adjust’ and ‘Observe’ Tools.

2.2 Driving a RC Servomotor with PWM

RC servos can be either analog or digital. Type of the servo 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 $+90^\circ$ 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 servo in relationship with its position can be seen in Figure 2.3.

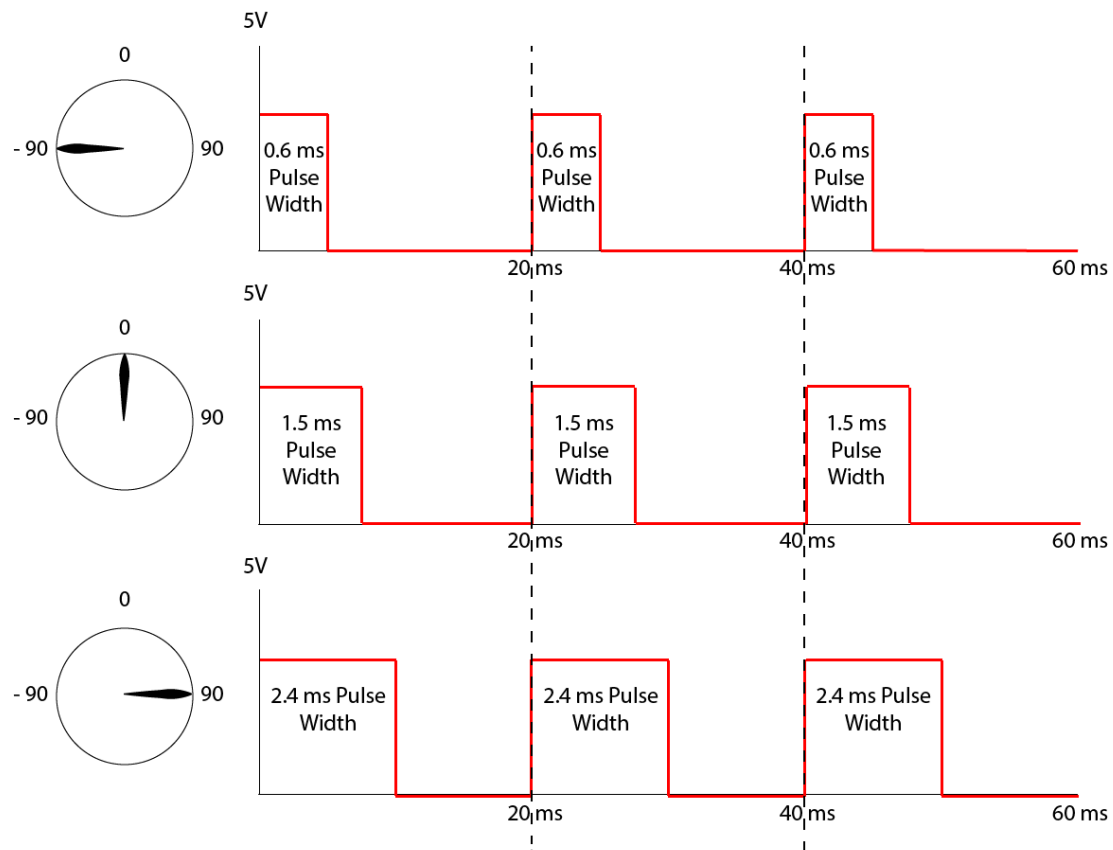


Figure 2.3: PWM Signal and Servo Position Relation

3 Filtering

There are two main reasons of derivative ripples. One of them is measurement noise. While the data is collected from sensor, other signals may distort measured data and high frequency measurement noise occurs. This difference from the actual data results in miscalculations and misbehaving on the system. The other reason is the discrete time derivative. Loop time is very small and even when small changes in measured data are divided by the loop time, so the derivative of them is high value, it causes many ripples. Therefore, a filter must be used to get slower and to reduce ripples. If the filter's parameters are small enough values, the system excessively slows down. On the contrary, if they are high enough values, derivative ripples continue to occur. Consequently, "30" is good value for the filter. Also, another filter is necessary for derivative so "20" is determined as enough value. The ripples continue but it is sufficient to control the system. Transfer functions for the derivative filter and filter are below:

$$Filter(s) = \frac{30}{s + 30} \quad Derivative\ Filter(s) = \frac{20}{s + 20}$$

3.1 In-Lab Exercise

1. Open up the "Potentiometer Sensor and Filtering.vi".
2. You will see the following front panel and the block diagram shown below:

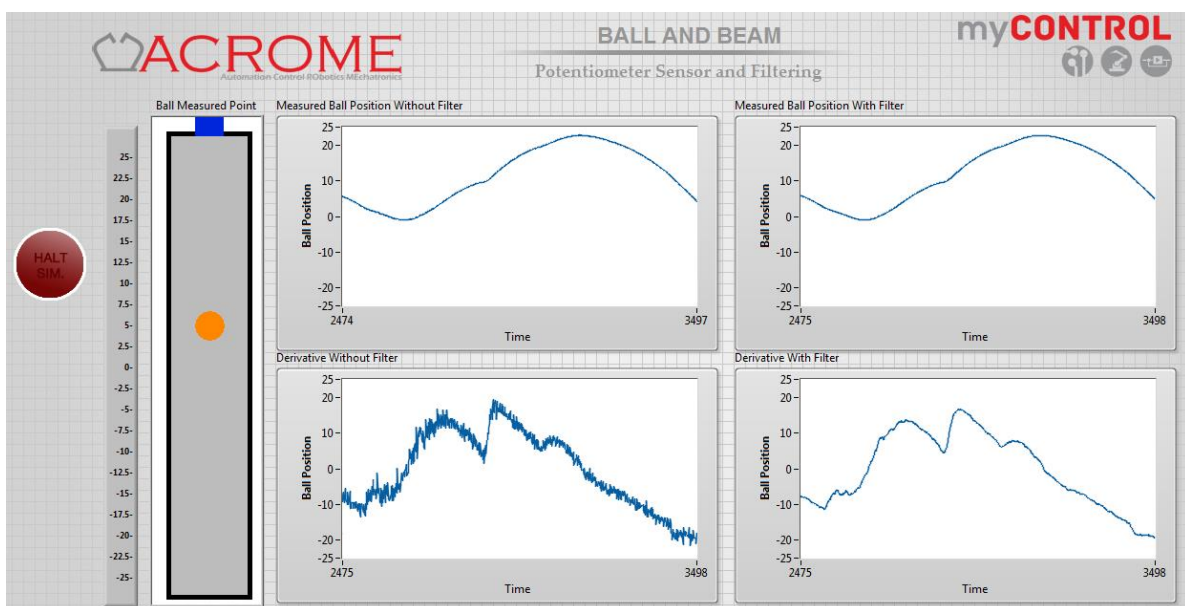


Figure 3.1: Front Panel of "Potentiometer Sensor and Filtering.vi"

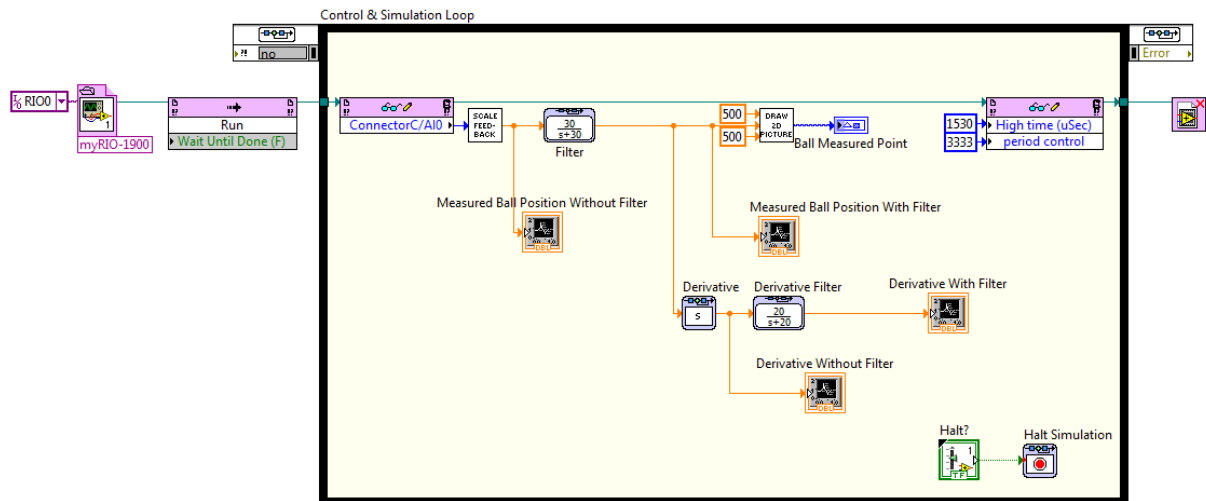


Figure 3.2: Block Diagram of “Potentiometer Sensor and Filtering.vi”

The VI only runs when there is a touch input on the panel. On the right; a 2D representation of the touch panel can be seen. The black circle shows where the touch occurs. Non-Filtered and Filtered Derived Position Data charts can be seen at the middle. Finally on the right; filter parameters can be seen. It is pre-configured to the optimal value mentioned above.

3. Touch the panel and notice that black circle moving along with your touch. This is your position feedback.
4. While touching the panel, observe the charts. As you can see non-filtered derived position data has more *spikes* or *ripples* which are need to be ruled out before any calculation. Notice that the filtered data is closer to what we want, with reduced spikes and ripples.
5. Go ahead and change filters parameters. How will filter and the system react with different parameters?

4 System Modelling

The physical model of the Ball and Beam is shown in Figure 4.1. According to the physical model, servo motor is attached on the base of the system at the end of the beam. Servo motor is connected to the beam over link with rod and ball joint. Beam is connected to the base plate of the system over a cord joint with one degree of freedom. Potentiometer sensor is stucked on the beam. Ball moves on the potentiometer sensor in v-shaped aluminum beam. Whole system is also assembled on plexiglass base on center and end of the beam. System parameters are explained in Table 4.1 which are used for obtaining the mathematical model of the ball and beam system.

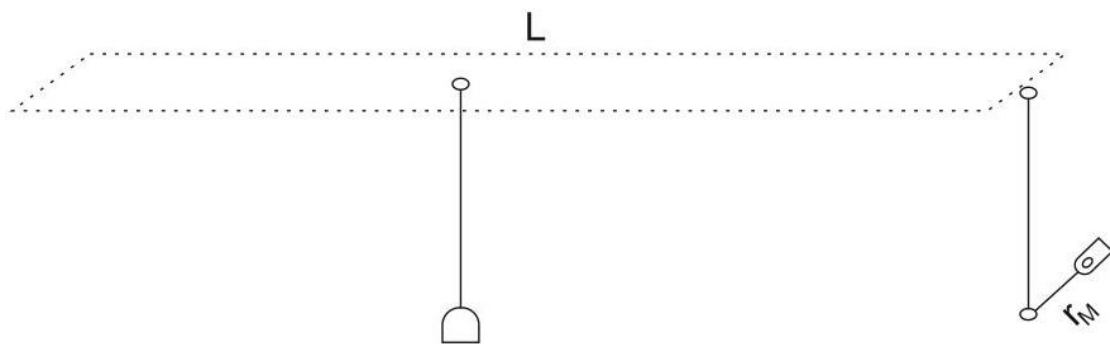


Figure 4.1: The Physical Model of the Ball and Beam

Symbols	Definition	Value	Unit
L	Beam Length	0.5	[m]
r_M	Motor Arm Length	0.035	[m]
r_b	Radius of the Ball	0.035	[m]
m_b	Mass of the Ball	0.175	[kg]
J_b	Rotational Inertia of the Ball	0.00008575	[kg * m ²]
g	Acceleration due to Gravity	9.81	[m/s ²]
α	Beam Angle	(Variable)	[degrees]
ϑ	Motor Angle	(Variable)	[degrees]

Table 4.1: Ball and Beam Parameters

Before to start modeling of the system, following assumptions have to be considered:

Ball and beam is in contact under any circumstances and the ball rolls on the beam without slipping during the obtaining the model of the system.

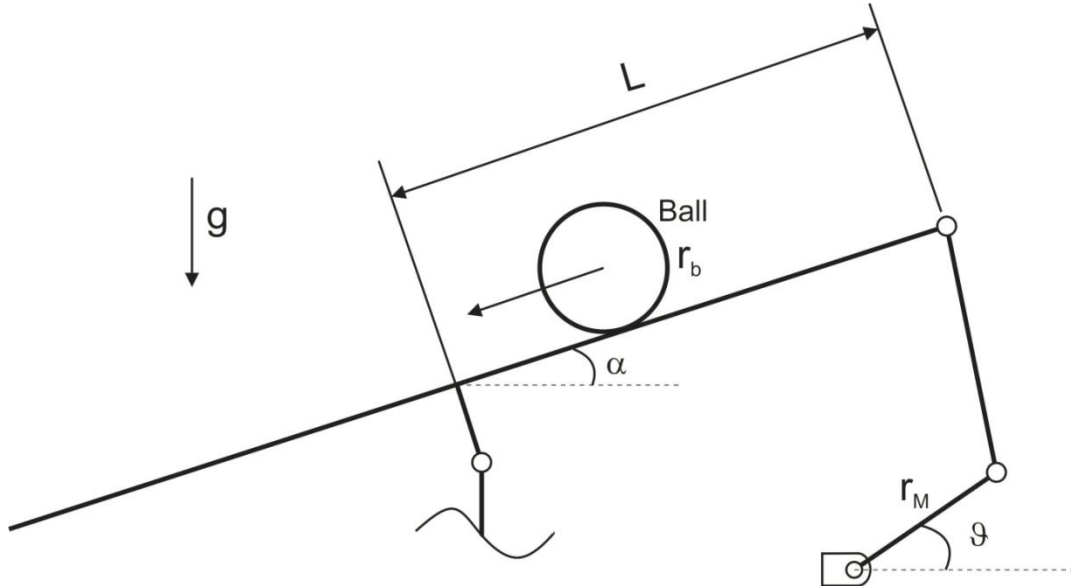


Figure 4.2: Free Body Diagram

4.1 Non-Linear Equation of Motion

The first step of the modelling is to derive the dynamic equation of motion. The equation of motion describes the relationship between motion of the ball (x_b) and the angle of the beam (α). In addition, velocity-dependent friction forces are not neglected because accurate model of the ball and beam system cannot be obtained.

In the following sections, the equation of motion will be derived with two different methods such as Lagrangian Method and Newton's Law of Motion.

4.1.1 Lagrangian Method

The Lagrangian Method derives the equation of motion through the relation of the kinetic and potential energy of the system. The Lagrangian Equation:

$$\frac{\partial}{\partial t} \left(\frac{\partial L}{\partial \dot{x}} \right) - \frac{\partial L}{\partial x} = F_{x,f} \quad (4.1)$$

where $F_{x,f}$ is the friction force which depends on velocity of the ball. Now, we will calculate Lagrangian (L) to obtain the terms of Lagrangian Equation.

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

$$L = E_{kin} - E_{pot} \quad (4.2)$$

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

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

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

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

$$E_{kin,R} = \frac{1}{2}J_b \omega_b^2 = \frac{1}{2}J_b \frac{v_b^2}{r_b^2} = \frac{1}{2}J_b \frac{(\dot{x}_b^2)}{r_b^2} \quad (4.5)$$

Thus,

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

The potential energy of the ball for given beam angle can be described as:

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

Therefore the Lagrangian is equal to:

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

The partial derivative terms of the Equation 4.1 are as follows:

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

And partial derivative of L with respect to position of the ball (x_b):

$$\frac{\partial L}{\partial x_b} = m_b g \sin(\alpha) \quad (4.10)$$

k_f is the coefficient of viscous friction. Thus, right hand side of the Lagrangian equation becomes:

$$F_{x,f} = k_f \dot{x}_b \quad (4.11)$$

So the differential equation of the motion becomes:

$$\left(m_b + \frac{J_b}{r_b^2}\right) \ddot{x}_b - m_b g \sin(\alpha) = k_f \dot{x}_b \quad (4.12)$$

Solving the equation for acceleration and velocity of the ball the required form of the equation is derived:

$$\ddot{x}_b - \frac{k_f r_b^2}{m_b r_b^2 + J_b} \dot{x}_b = \frac{m_b g r_b^2}{m_b r_b^2 + J_b} \sin(\alpha) \quad (4.13)$$

4.1.2 Newton's Law of Motion

The sum of forces acting on the ball equals according to the Newton's Law of Motion:

$$\sum F = m_b \frac{d^2 x_b}{dt^2} \quad (4.14)$$

The forces acting on the ball can be described as:

$$\sum F = F_{x,t} - F_{x,r} - F_{x,f} \quad (4.15)$$

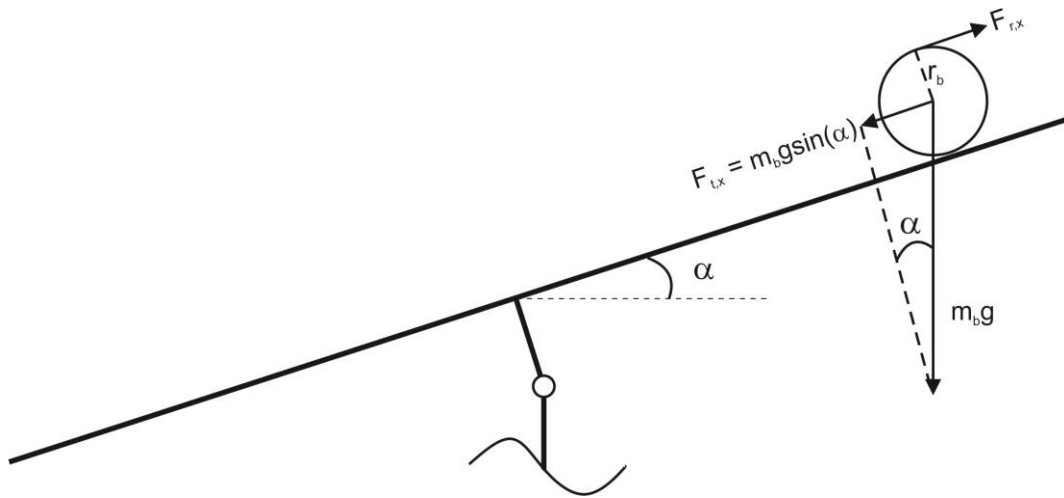


Figure 4.3: Free Body Diagram

Where $F_{x,t}$ is the translational force acting on the ball due to gravity and inclined plane as shown Figure 4.3. There is also velocity dependent friction force acting on the ball ($F_{x,f}$) is opposite direction of the translational force.

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

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

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

Thus,

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

By solving the equation for acceleration and velocity of the ball the required form of the equation is derived:

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

As it can be easily seen from the Equation (4.1) and (4.18), the derived equations of motion for both methods are same, if the same friction force is added as a non-conservative force.

4.2 Modelling of Ball and Beam System

The aim of modelling is to derive the physical relationship between the output and input of a system. In our case, the output of the system is the position of the ball (x_b). The input of the system is the motor angle (θ) as it can be seen in Figure 4.4.

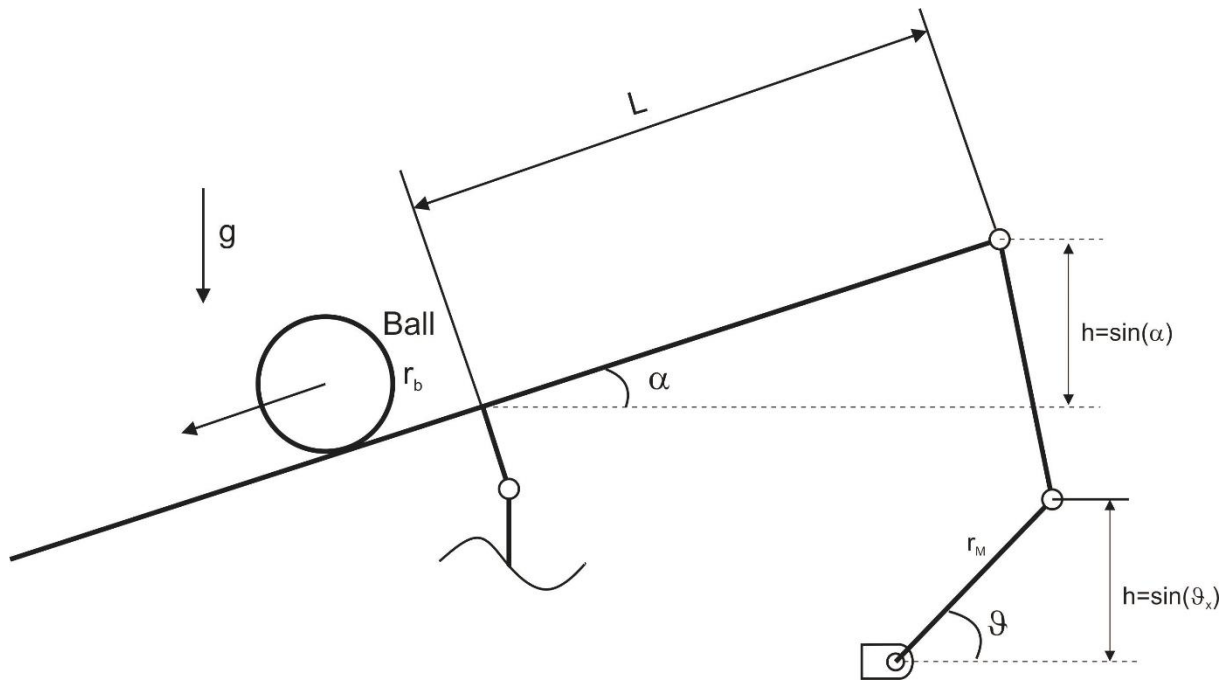


Figure 4.4: Relationship between Motor Angle and Beam Angle

Figure 4.4 also illustrates that:

$$\sin(\theta)r_M = \sin(\alpha)L = h \quad (4.20)$$

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

$$\sin(\alpha) = \frac{r_M}{L} \sin(\theta) \quad (4.21)$$

We can write instead of $\sin(\alpha)$ term in the Equation 4.19 the right side of the equation 4.21. Consequently we have the differential equations of the system which describes the relationship between the output and input of the system:

$$\ddot{x}_b - \frac{k_f r_b^2}{m_b r_b^2 + J_b} \dot{x}_b = \frac{m_b g r_b^2 r_M}{(m_b r_b^2 + J_b) L} \sin(\theta) \quad (4.22)$$

4.2.1 Linearization around Operating Point

To obtain the transfer function of the system, the differential equation has to be linearized about the operating point ($x = 0$). For small angles:

$$\sin(\theta) \approx \theta \quad (4.23)$$

Thus left side and right side of the Equation 4.22 can be rewritten as:

$$\ddot{x}_b - \frac{k_f r_b^2}{m_b r_b^2 + J_b} \dot{x}_b = \ddot{x}_b - K_{FF} \dot{x}_b \quad , \quad \frac{m_b g r_b^2 r_M}{(m_b r_b^2 + J_b) L} \vartheta = K_{BB} \vartheta \quad (4.24)$$

By using the value of system parameters which were listed in Table 4.1, K_{BB} and K_{FF} can be easily calculated.

4.2.2 Obtaining Transfer Function of Plant

The plant's differential equation is:

$$\ddot{x}_b - K_{FF} \dot{x}_b = K_{BB} \vartheta \quad (4.25)$$

Taking the Laplace transform we obtain as below:

$$s^2 X_b(s) + K_{FF} s X_b(s) = K_{BB} \vartheta(s) \quad (4.26)$$

The transfer function can be derived the Equation 4.27:

$$G_{BB}(s) = \frac{X_b(s)}{\vartheta(s)} = \frac{K_{BB}}{s^2 + K_{FF} s} = \frac{0.53}{s^2 + 0.25s} \quad (4.27)$$

4.3 Modelling of Actuator

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

$$G_M(s) = \frac{\vartheta(s)}{V_m(s)} = \frac{K_M}{\tau s + 1} \quad (4.28)$$

K_M is the gain and τ is the time constant of the actuator. K_M is calculated as 2 and τ is calculated as 0.01. After calculations, motor transfer function is obtained as below:

$$G_M(s) = \frac{2}{0.01s + 1} \quad (4.29)$$

4.4 Cascade Control of Ball and Beam

Ball and beam has cascaded structure which includes inner and outer loop as shown in Figure 4.5.

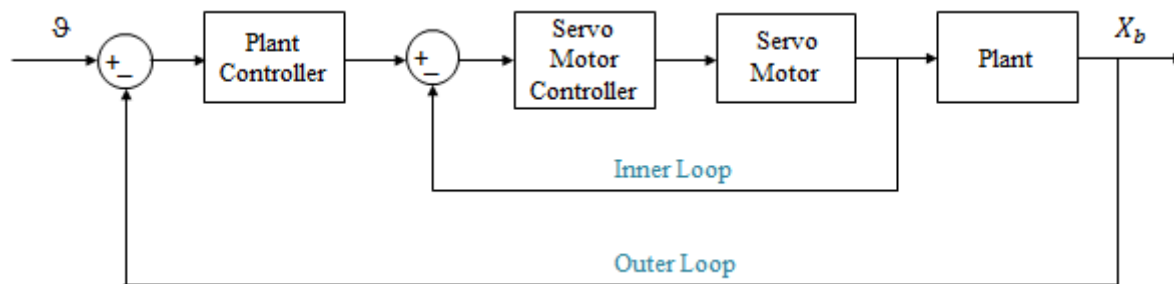


Figure 4.5: Block Diagram Ball and Beam

Inner loop is unity feedback closed loop system and controls the servo motor position. Servomotor of the ball and beam automatically controls its position thanks to own controller. Inner loop reaches the steady state in a split second. Therefore inner loop transfer function can be obtained first order transfer function for ball and beam. When the inner loop is reduced as actuator, the new block diagram is as below:

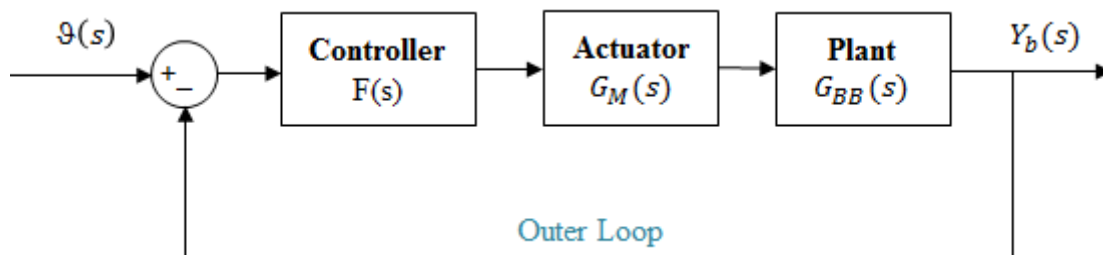


Figure 4.6: Block Diagram of System after Reduction

Plant's and Actuator's transfer function were obtained in the previous section. When the parameters are replaced, block diagram is also shown as below:

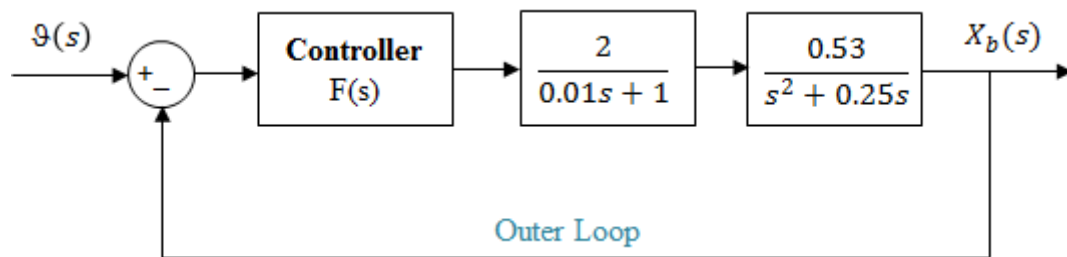


Figure 4.7: Block Diagram of Outer Loop

Outer loop consists of plant, actuator and their controller. Designing of the controller will explain in Chapter 7.

5 System Identification

In this chapter, a new modelling method called system identification method is used for obtaining the transfer function. An identification procedure consists of generating a suitable input data, measuring the system response corresponding to input data and finally using the collected data to obtain model of the dynamical system. There are three identification approaches such as white-box, gray-box and black-box. White-box which estimates parameters of a physical model from data is used for ball and beam system.

5.1 Collecting Estimation Data

Data acquisition is the most critical point for accurate plant modelling so we firstly acquire the input and output data from the physical system. Input data is angle of the beam and output data is position of the ball. Beam angle is changed with the help of slider in respect to ball position because the open loop system is unstable. Input and output data are also called the training or estimation data which is collected via “Collecting Estimation Data.vi”.

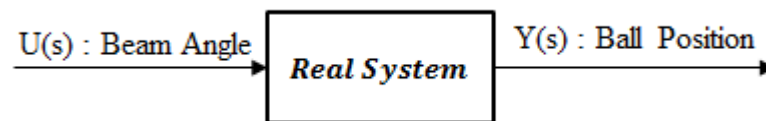


Figure 5.1: Block Diagram of the Open Loop System

5.1.1 In Lab Exercise:

1. Open the “Collecting Estimation Data.vi” is shown in Figure 2.
2. Run the vi. Click the “ACTION” button
3. Generate the input data with using slider on the front panel. (Be careful not to hit edge of the beam!)
4. Observe the ball position shown in graph. If the collecting data is suitable jump the next step, or else return the first step.
5. Halt the simulation and close the vi.

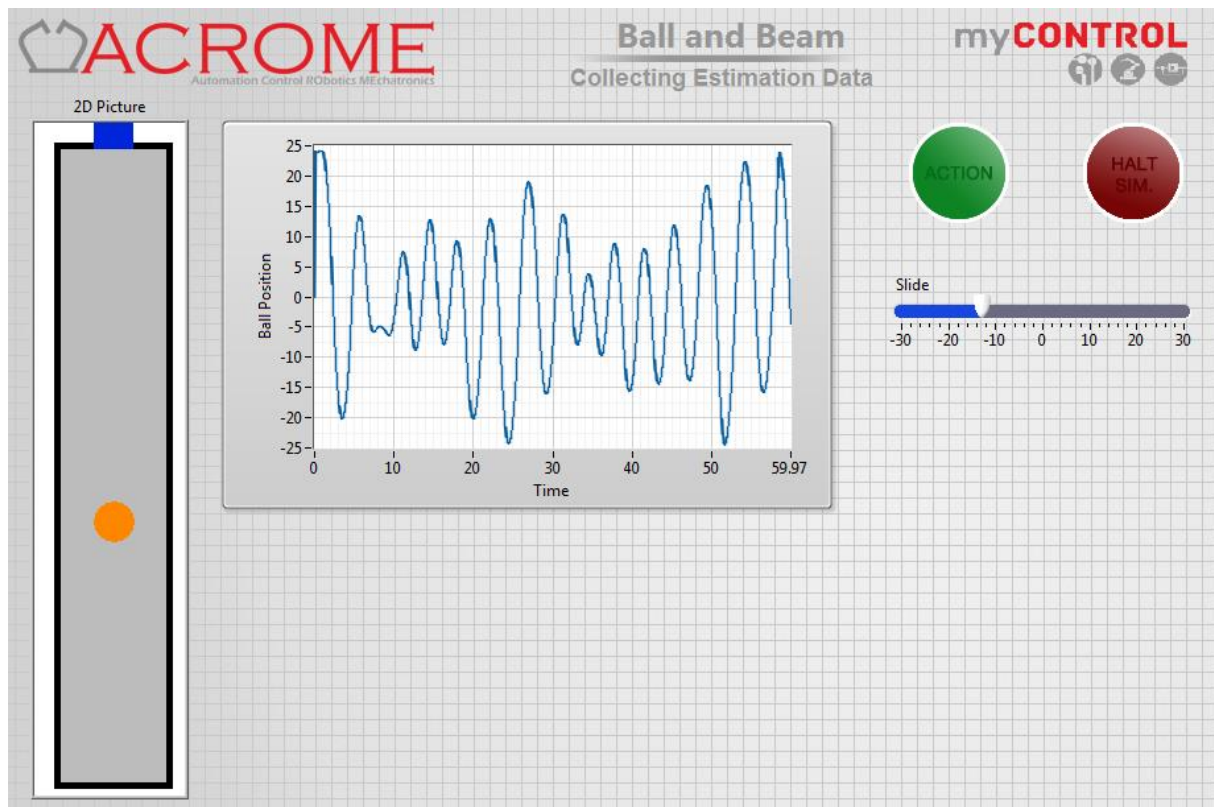


Figure 5.2: Front Panel of "Collecting Estimation Data.vi"

5.2 Obtaining Estimated Model via System Identification

To obtain estimated model, "System Identification.vi" is opened and determined the settings of model order and cursor position. The final stage of identification is controlling model validation. Validation data applied to system and model at the same time. Then, model and system response which are on the same validation graph can be compared.

The estimated model contains motor and system dynamics is as below when degree of the system is chosen two and number of poles at origin is chosen zero. The following transfer function is also used for design of the controllers.

$$G_{BB}(s) = \frac{22.5}{28.3s^2 + 16s + 1}$$

5.2.1 In Lab Exercise:

1. Open the "System Identification.vi" is shown in Figure 3.
2. Determine the cursor position and settings of the model order.
3. Run the vi.

4. Observe model and real system responses which are shown in Model validation graph.
5. Save the transfer function to use design of the controller.
6. Halt the simulation and close the vi.

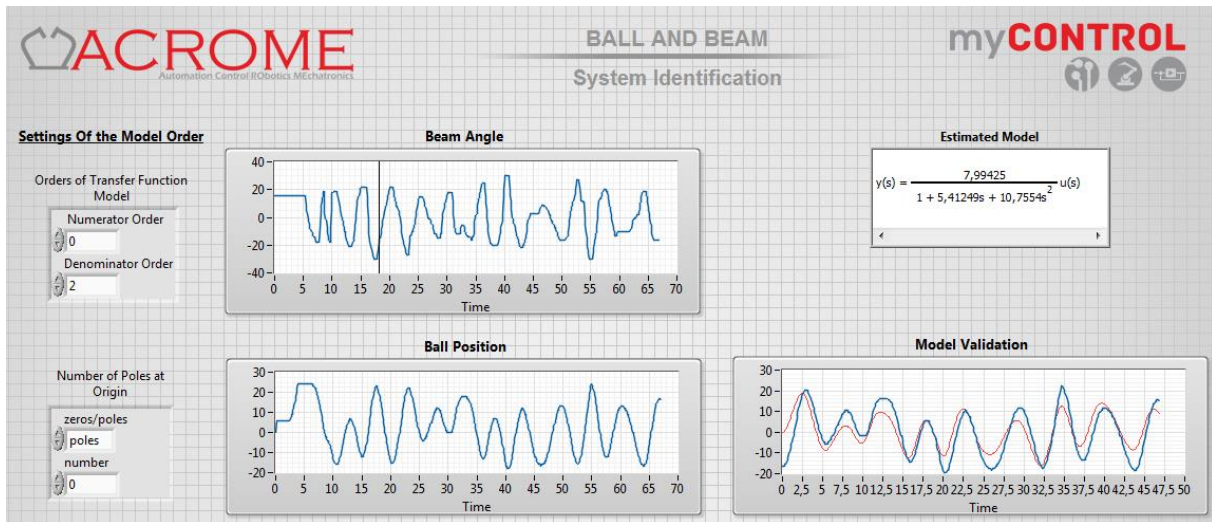


Figure 5.3: Front Panel of "System Identification.vi"

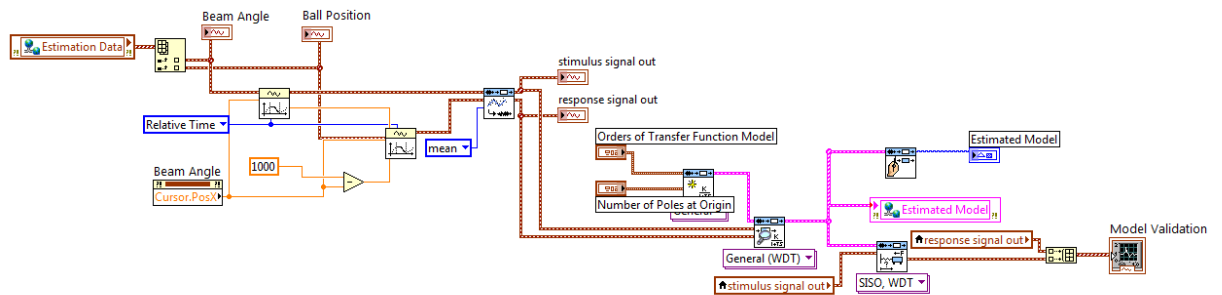


Figure 5.4: Block Diagram of "System Identification.vi"

6 Performance Measures

Performance measures or criteria are defined for second order systems which is applied step input. Different performance measures are determined considering the systems. Depending on performance measures, damping ratio (ξ) and natural frequency (ω_n) are found. Also, if damping ratio and natural frequency are known, performance measures can be calculated. In this section, all parameters are investigated.

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

6.1.1 Damping Ratio (ξ)

When step input is applied, the system gives a response depending on the damping ratio. Damping ratio and the system response coupling can be seen in Table 6.1. By using percentage overshoot (PO), damping ratio can be calculated.

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

6.1.2 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.

6.1.3 Percentage Overshoot

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

$$\begin{aligned} \text{Percentage Overshoot} &= 100 * \frac{\text{Maximum Output Value} - \text{Final Output Value}}{\text{Final Output Value}} \\ &= 100 * e^{\frac{-\xi\pi}{\sqrt{1-\xi^2}}} \end{aligned}$$

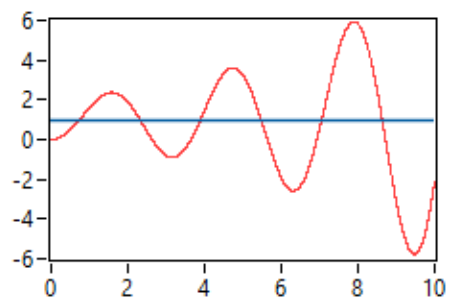
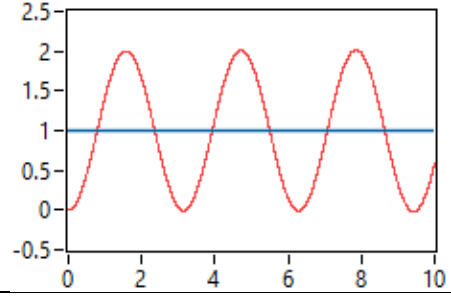
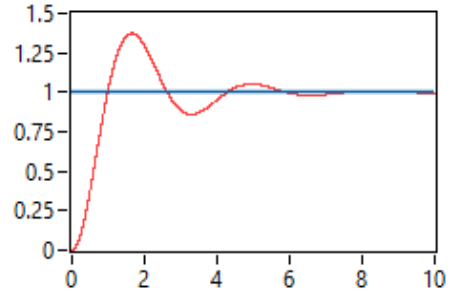
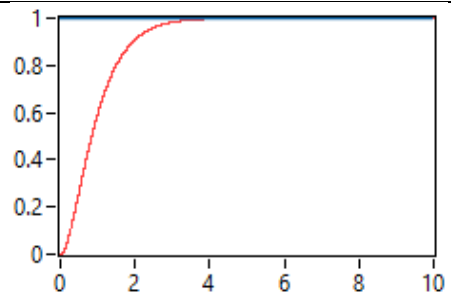
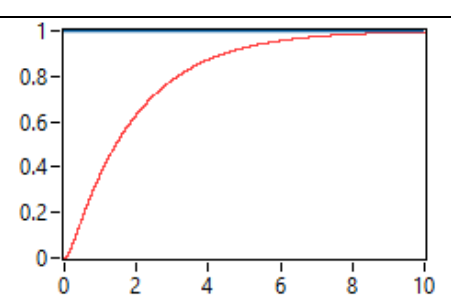
Damping Ratio (ξ)	System Behaviour	System Input and Output
$\xi < 0$	Unstable	
$\xi = 0$	Undamped	
$0 < \xi < 1$	Underdamped	
$\xi = 1$	Critically Damped	
$1 < \xi$	Over damped	

Table 6.1: Damping Ratio, system behaviour and system response graph

6.1.4 Peak Time (t_p)

Peak time is the elapsed time from applying the step input until it is reaching the maximum value. Also, peak time can be calculated with the help of the damping ratio and the natural frequency.

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

6.1.5 Settling Time (t_s)

Settling time is the time when the system enters 2% or 5% band of system response final value. 2% band will be used from now on. Also, there is a formula for calculating settling time as below:

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

6.1.6 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. It is defined as steady state error. If the system has a steady state error, it can be removed easily with the help of an integral type of controller.

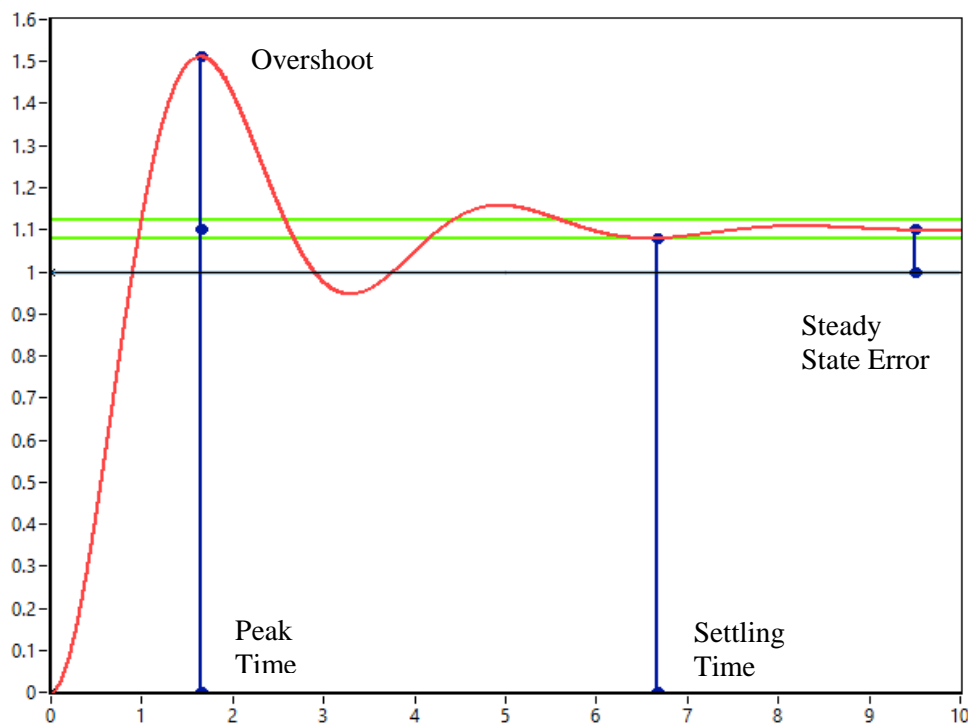


Figure 6.1: Overshoot, settling time, peak time and steady state error

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.

6.1.7 In-Lab Exercise

1. Open “Performance Measures.vi” as follows,

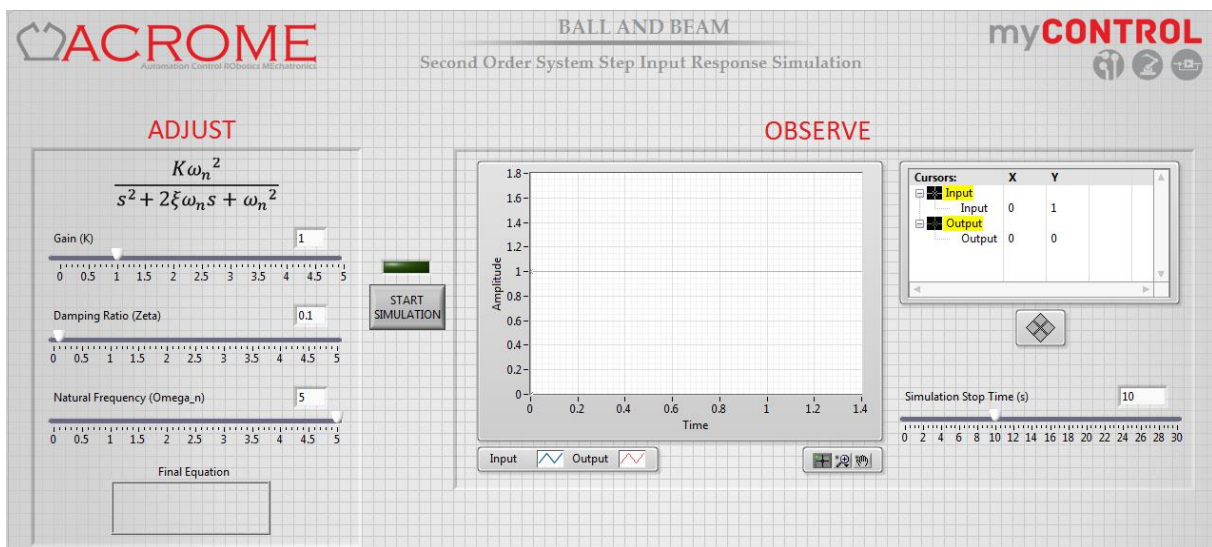


Figure 6.2: Front Panel of Performance Measures.vi

2. Enter the values as 1.1, 0.3, 2 for K, ξ and ω_n respectively (NOTE: Unit step input is applied to the system every time),
3. Calculate the following performance measures: percentage overshoot, settling time, peak time and steady state error,
4. Run the VI, observe the system input and output on the graph and determine the performance measures,
5. Compare the results were found in Step 3 and Step 4 with each other; are they consistent or not?
6. Enter different values for K, ξ and ω_n , and then observe the change in performance measures.

7 Control System Design

Open loop systems are called the manual control systems as the output has not affected upon 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 system and they have many advantages like disturbance rejection, noise attenuation, presence of nonlinearity and robustness, so closed loop system are preferred in control applications in general.

The purpose of the controller design is with closed loop system, the previously specified performances measures can be achieved. Choosing type of 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 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 also 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 a non-interacting algorithm. Academic and parallel forms of PID controllers and fuzzy controller will be used for ball and beam position control. Also, other types of controller such as P, PI and PD are obtained by using academic and parallel forms of PID.

7.1 Academic PID controller

Academic PID controller is mathematically expressed as:

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

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

7.2 Parallel PID controller

Parallel PID controller is mathematically expressed as:

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

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 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.

7.3 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).

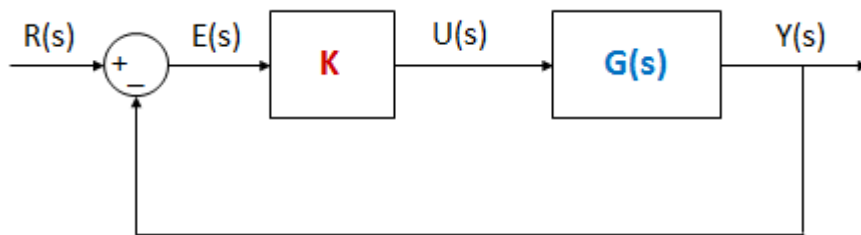


Figure 7.1: General Block Diagram of Root Locus

To draw a root locus, firstly open loop transfer function 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 the characteristic polynomial. Characteristic polynomial is a function of K as shown below:

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

Finally, loci 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 analyze. Root locus is also used for controller design, but our controller will not be designed using the root locus.

7.4 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:

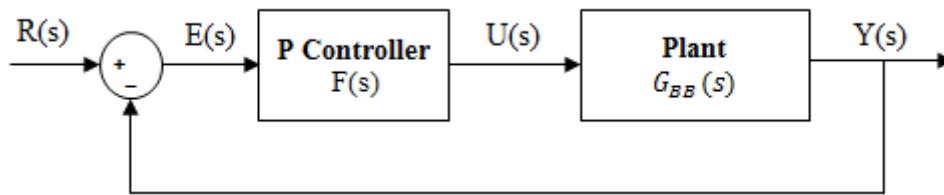


Figure 7.2: General Block Diagram with P Controller

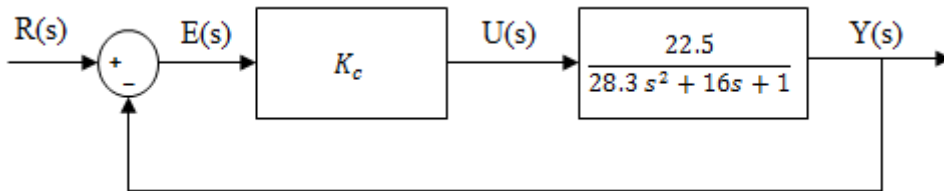


Figure 7.3: The Closed Loop System with P Controller

7.4.1 Sample Design with P Controller

We will design the controller as P-type. Transfer functions of controller and plant are below respectively:

$$F(s) = K_c, \quad G(s) = \frac{22.5}{28.3s^2 + 16s + 1}$$

We obtain the closed loop transfer function depending on Figure 7.3.

$$T(s) = \frac{Y(s)}{R(s)} = \frac{F(s) * G(s)}{1 + F(s) * G(s)} = \frac{22.5K_c}{28.3s^2 + 16s + 1 + 22.5K_c}$$

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

$$P_c(s) = 28.3s^2 + 16s + 1 + 22.5K_c$$

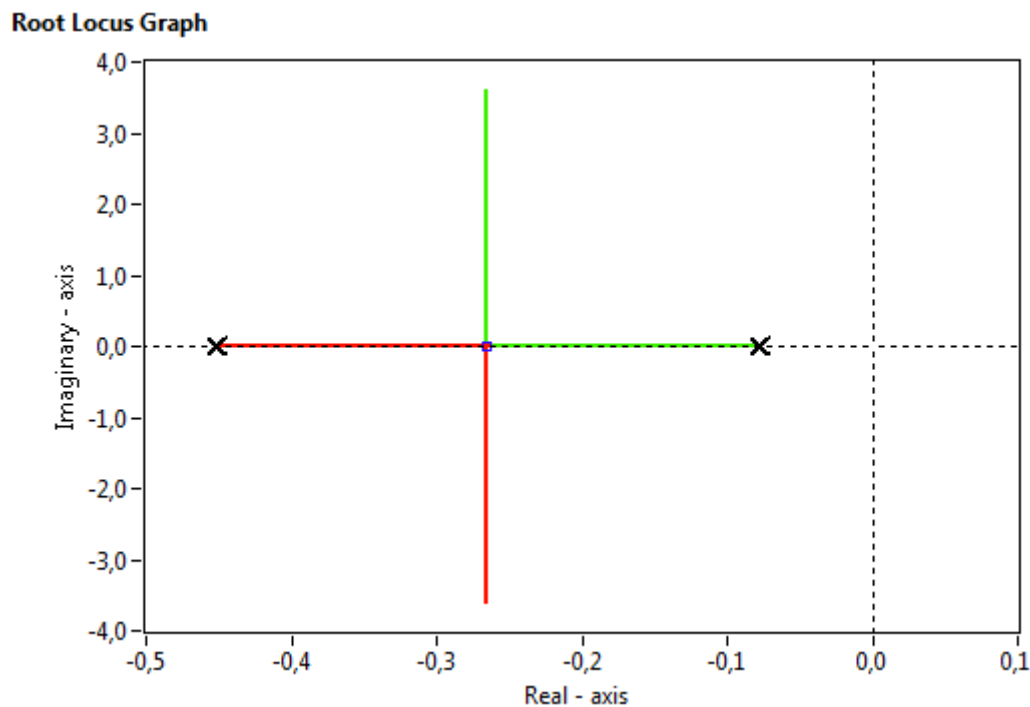


Figure 7.4: Root Locus Graph of Closed Loop System with P Controller

The root locus of the open loop system is drawn in Figure 7.4. If we analyze the root locus, there are two real poles or complex conjugate pole pair of the closed loop system corresponding to value of K_c .

For a linear system to be stable, all poles must have negative real parts. Real part of the all poles is always negative for each value of K_c so we expect we expect to find that the closed system is unstable. However, the real system response is unstable because of nonlinearities such as delay time and friction for greater than $K_c = 3.5$. On the other hand, model response is stable for all K_c because of obtaining linear model. As it was expected, the model and real system response is consistent for small value of K_c . The step responses model and real system is shown for $K_c = 1.3$ as below:

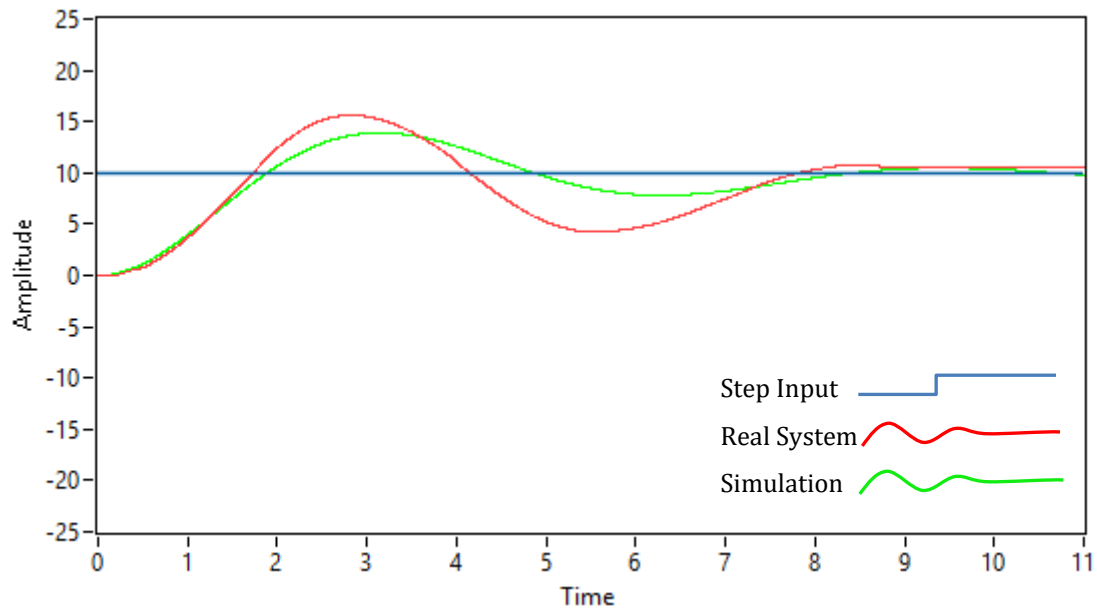


Figure 7.5: The Real-Time, Simulation and Step Input Graphs with P Controller

7.4.2 In-Lab Exercise

7.4.2.1 Control & Simulation Loop Configuration

Before the exercise, some configurations are necessary for Control & Simulation Loop. Same configurations will be applied to Control & Simulation Loop.

We must right click to Control & Simulation Loop, and select “Configure Simulation Parameters...” selection. Configurations in popup window will be in the Figure 7.6 shown as below. Finally, we will click to the “OK” button.

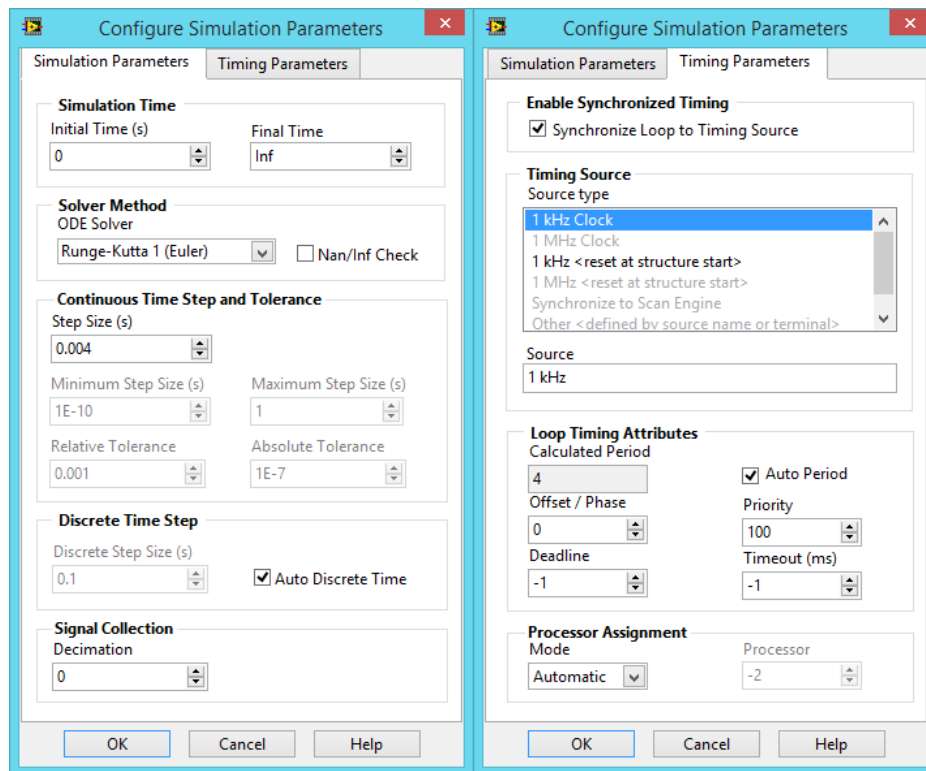


Figure 7.6: Configuration Simulation Parameters

7.4.2.2 In-Lab Exercise

Answer the following questions for the block diagram as shown Figure 7.3. Sample design of P controller was made for a specific performance measures. We expect to solve In-Lab Exercise for different values.

1. Determine the order of the open loop system without a controller.
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 unit step is applied?
6. Is the closed loop system stable? If not, can it be stable?
7. Try to your controller into real system and simulation. Open "PID Controller Design.vi" and enter T_d is "0" and T_i is "Inf" for obtaining P-controller.

7.5 PD Controller

PD controller is generally used for position control applications because of the plant's integrator. It approaches two of closed loop system poles to desired locations and adds a zero at the same time while it does not change 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 a split second. This pulse is described as a derivative kick. Design of the PD controller on the forward path results in 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:

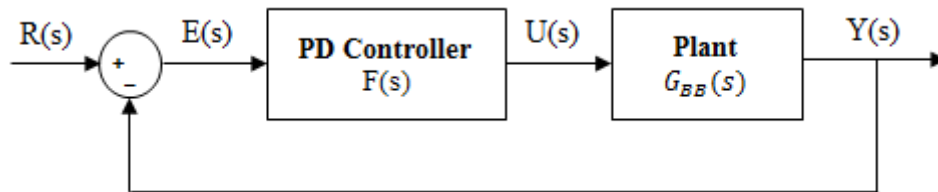


Figure 7.7: General Block Diagram of the Closed Loop System with PD Controller

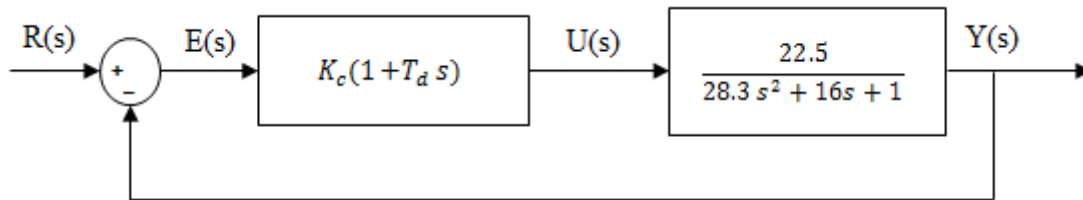


Figure 7.8: Closed Loop System with PD Controller

7.5.1 Sample Design with PD controller

We will design a PD controller that satisfies 35% percentage overshoot and 8 seconds settling time with no steady state error. Firstly, the closed loop transfer function and its characteristic polynomial are obtained in Figure 7.8:

$$T(s) = \frac{Y(s)}{R(s)} = \frac{F(s) * G(s)}{1 + F(s) * G(s)} = \frac{22.5K_c(1 + T_d s)}{28.3s^2 + 16s + 1 + 22.5K_c(1 + T_d s)}$$

$$P_C(s) = 28.3s^2 + 16s + 1 + 22.5K_c(1 + T_d s)$$

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

$$\xi = \frac{-\ln(PO)}{\sqrt{\pi^2 + (\ln(PO))^2}} = \frac{-\ln(0.35)}{\sqrt{\pi^2 + (\ln(0.35))^2}} = 0.317$$

$$\omega_n = \frac{4}{\xi * t_s} = \frac{4}{0.317 * 8} = 1.578$$

Thirdly, the desired characteristic polynomial is determined:

$$P_D(s) = a * (s^2 + 2\xi\omega_n s + \omega_n^2) = a * (s^2 + s + 2.489)$$

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

$$a = 28.3 \quad K_c = 3.08 \quad T_d = 0.177$$

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 36% and t_s equals to 6.9 seconds.

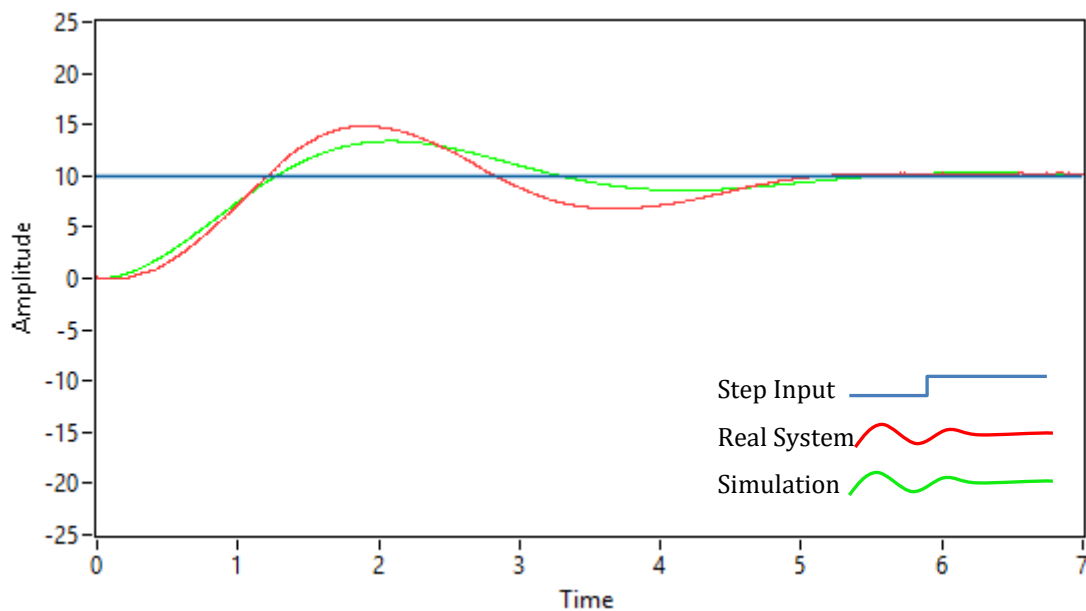


Figure 7.9: The Real-Time, Simulation and Step Input Graphs with PD Controller

7.5.2 In-Lab Exercises

Answer the following questions for the block diagram as shown Figure 7.8.

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 $OS \leq 10\%$ and $t_s \leq 5$ seconds).
5. Obtain the desired characteristic polynomial with the help of Step 4 (Does residue polynomial become a necessity for PD controller?).
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 via "PID Controller Design.vi" as T_i is chosen "Inf". 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 the response of simulation and real system via "PID Controller Design.vi". Are they consistent?

7.6 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 the 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. Most of the time, P type controller is not sufficient; therefore, root locus is easily changed with a derivative term. Block diagrams are shown below:

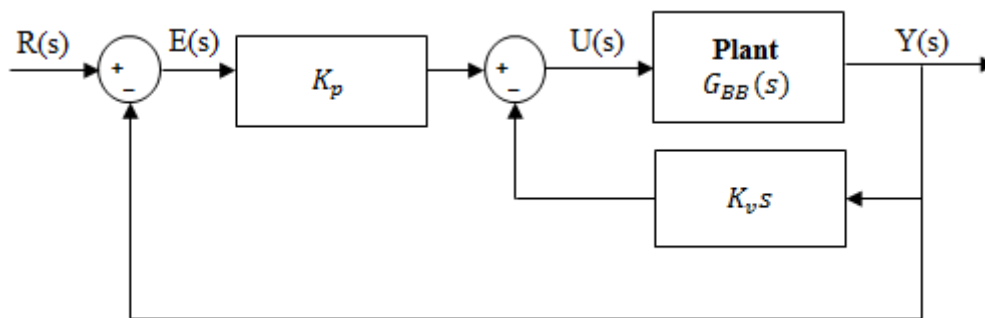


Figure 7.10: General Block Diagram of Closed Loop System with PV Controller

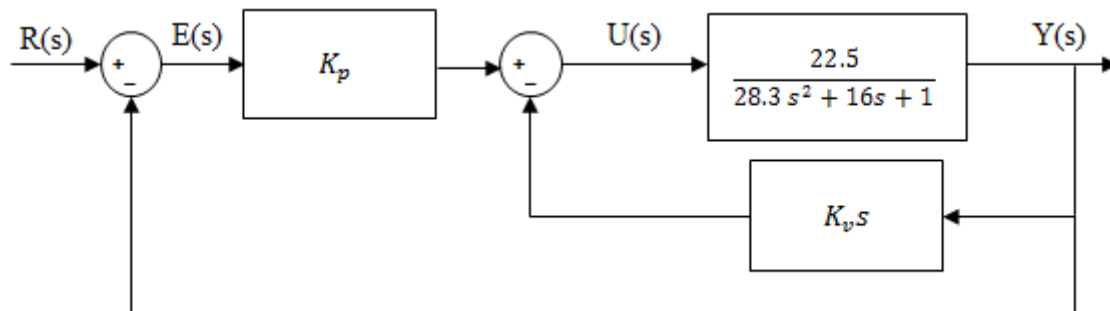


Figure 7.11: Closed Loop System with PV Controller

7.6.1 Sample Design with PV controller

We will design a PV controller with respect to 25% percentage overshoot, 8 seconds settling time and no steady state error. Gain part of the PV controller on the forward path represents $F_1(s)$ and derivative part of the PV controller on feedback path represents $F_2(s)$. The inner loop transfer function is obtained and is shown below:

$$T_{inner}(s) = \frac{G_{BB}(s)}{1 + F_2(s) * G_{BB}(s)} = \frac{22.5}{28.3s^2 + 16s + 1 + 22.5K_v s}$$

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

$$T_{outer}(s) = \frac{F_1(s) * T_{inner}(s)}{1 + F_1(s) * T_{inner}(s)} = \frac{22.5K_c}{28.3s^2 + 16s + 1 + 22.5K_v s + 22.5K_c}$$

PV controller is practically a PD controller because the characteristic polynomials of PV and PD controllers are the same.

$$P_c(s) = 28.3s^2 + 16s + 1 + 22.5K_c + 22.5K_v s$$

First of all, ξ and ω_n are calculated.

$$\xi = \frac{-\ln(PO)}{\sqrt{\pi^2 + (\ln(PO))^2}} = \frac{-\ln(0.25)}{\sqrt{\pi^2 + (\ln(0.25))^2}} = 0.401$$
$$\omega_n = \frac{4}{\xi * t_s} = \frac{4}{0.401 * 8} = 1.2385$$

Secondly, second order desired characteristic polynomial is obtained.

$$P_D(s) = a * (s^2 + 2\xi\omega_n s + \omega_n^2) = a * (s^2 + s + 1.534)$$

Finally, the unknown parameters, a , K_p and K_v , are found.

$$a = 28.3 \quad K_p = 1.88 \quad K_v = 0.55$$

Closed loop system has no steady state error. The settling time of the closed loop system is approximately 6.78 seconds and overshoot is 25%.

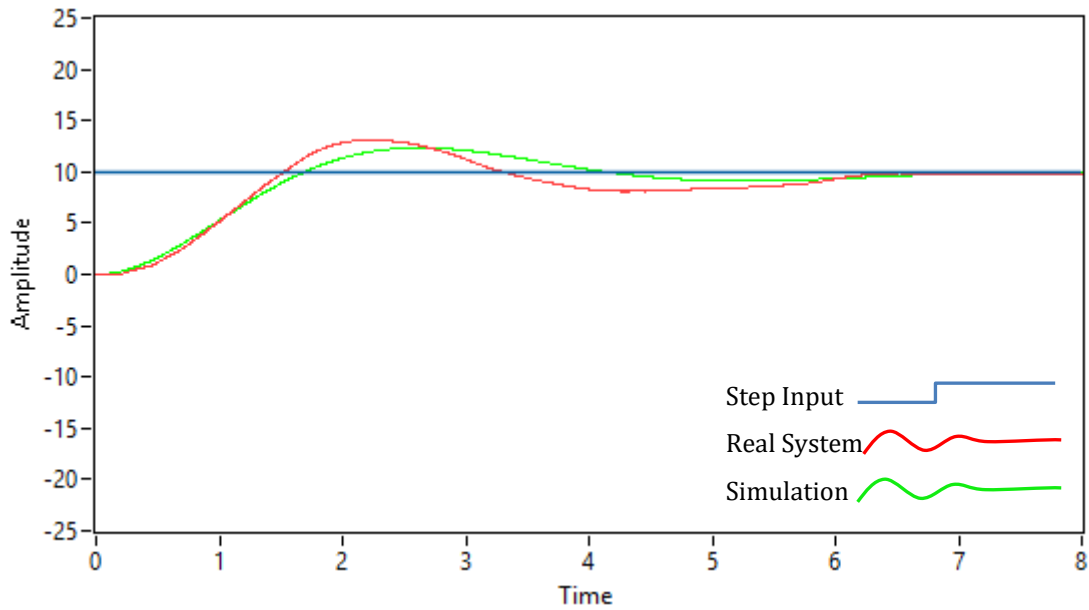


Figure 7.12: The Real-Time, Simulation and Step Input Graphs with PV Controller

7.6.2 In-Lab Exercises

Answer the following questions for the block diagram as shown Figure 6.11.

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 $\leq 10\%$ and $t_s \leq 5$ seconds).
5. Obtain the desired characteristic polynomial with the help of Step 4 (Does residue polynomial become necessity for PV controller?),
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 via "PV Controller Design.vi". 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 via “PV Controller Design.vi”. Discuss the differences the response of simulation and the real system. Are they consistent?

7.7 PID Controller

PID controller is the most popular controller even when using in non-linear applications. The types of non-linear controllers are complicated and are hardly implemented, so 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 system poles to a desired location. Nevertheless, like PI controller, PID controller does not usually return accurate result for position controller. The characteristic polynomial poles are stable in a small region within plant’s integrator poles. Block diagram with PID controller is shown below:

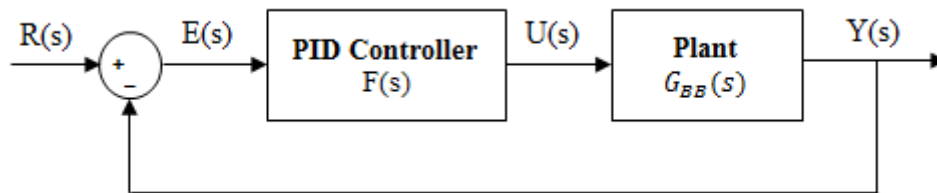


Figure 7.13: Closed Loop System with PID Controller

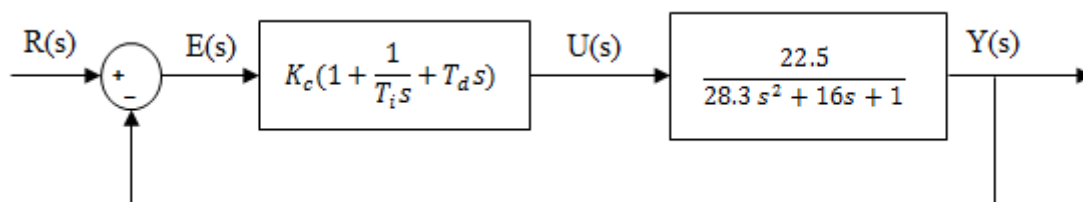


Figure 7.14: Closed Loop System with Academic PID Controller

When parallel PID parameters are increased independently, Table 7.1 shows the qualification results of the performance measures.

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

Table 7.1: Qualification Results Corresponding Increase in PID Parameters

7.7.1 Sample Design with PID controller

A PID controller will be designed that satisfies 50% percentage overshoot and 15 seconds settling time with no steady state error. Thanks to integrator, there is no steady state error. Firstly, closed loop transfer function and its characteristic polynomial are obtained in Figure 7.14:

$$T(s) = \frac{Y(s)}{R(s)} = \frac{F(s) * G(s)}{1 + F(s) * G(s)} = \frac{22.5K_c(T_i T_d s^2 + T_i s + 1)}{28.3T_i s^3 + 16T_i s^2 + T_i s + 22.5K_c(T_i T_d s^2 + T_i s + 1)}$$

$$P_C(s) = 28.3T_i s^3 + 16T_i s^2 + T_i s + 22.5K_c(T_i T_d s^2 + T_i s + 1)$$

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

$$\xi = \frac{-\ln(PO)}{\sqrt{\pi^2 + (\ln(PO))^2}} = \frac{-\ln(0.5)}{\sqrt{\pi^2 + (\ln(0.5))^2}} = 0.215$$

$$\omega_n = \frac{4}{\xi * t_s} = \frac{4}{0.215 * 15} = 1.2377$$

We find the controller parameters with equal coefficients of the designed and desired characteristic polynomials, so the order of both of them must be equal. If the order of the designed characteristic polynomial is higher than two, we add a multiplication polynomial described as the residue polynomial. For design of the PID 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. Residue polynomial is chosen as $(as + b)$, desired characteristic polynomial is determined:

$$P_D(s) = (as + b) * (s^2 + 0.533s + 1.532)$$

Finally, when desired and designed characteristic polynomials are equalized to each other a , b , K_c , T_i and T_d are calculated where T_i is the free parameter and it is chosen "1":

$$a = 28.3 \quad b = 42.41 \quad K_c = 2.89 \quad T_d = 0.639 \quad T_i = 1$$

Controller parameters are written for the closed loop system; simulation step response and real system response are shown in Figure 6.15.

Overshoot is 40.6% and settling time is 12.82 seconds. Closed loop system is satisfied our performance measures.

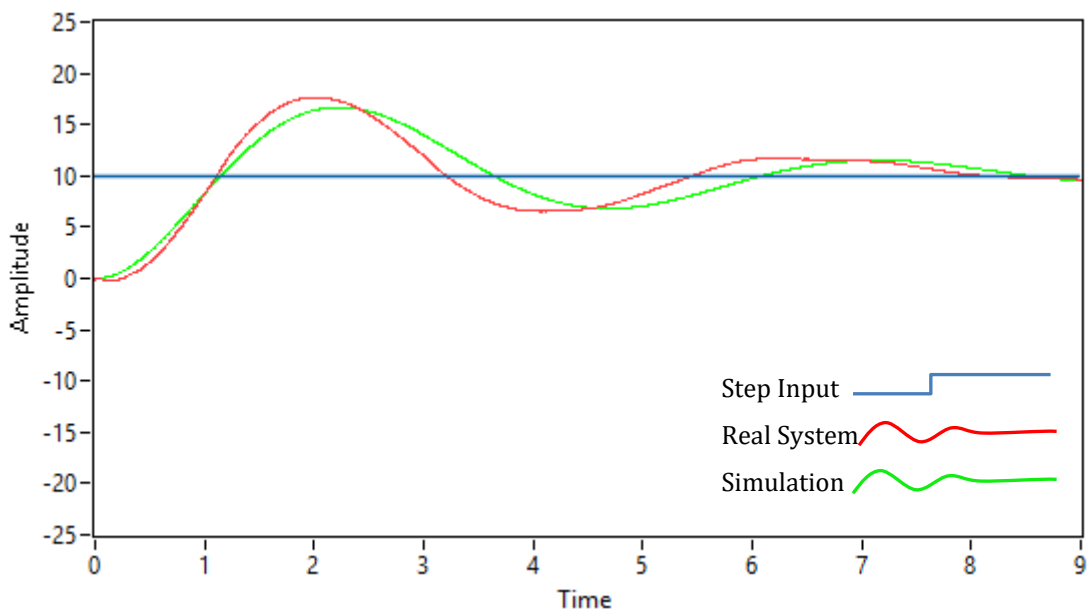


Figure 7.15: The Real-Time, Simulation and Step Input Graphs with PID Controller

7.7.2 In-Lab Exercises

Answer the following questions for the block diagram as shown Figure 7.14.

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 $OS \leq 20\%$ 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 via “PID Controller Design.vi”. 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 via “PID Controller Design.vi”. Discuss the differences the response of the simulation and the real system. Are they consistent?

7.8 Fuzzy Logic Controller

The concept of fuzzy logic was presented by Professor Lotfi Zadeh in 1965 and presented not as a control methodology. In 1972, he was presented a new paper and he was introduced the term *fuzzy rules* and *linguistic variables* in control theory. Today, fuzzy logic is used as a nonlinear problem-solving control system methodology. It is different from conventional control methods in terms of approach to solving problems. Conventional control methods assume that the behavior of the system is linear and the system transfer function is known. Hence, the controller parameters can be found mathematically by using the transfer function. In contrast to conventional control methods, fuzzy logic does not require a system transfer function or a linear system. Fuzzy logic uses linguistic variables such as negative, zero, positive, small, medium and large. These linguistic variables form fuzzy sets. Inputs and outputs of the fuzzy controller are formed by combining these fuzzy sets. Rule-base (linguistic rules) is obtained by using linguistic variables which are obtained previously. Fuzzy inference process is used for determining which rules are on to find which rules are relevant to the current station by using rule-base. Finally, a single output is produced by using one of the defuzzification methods.

7.8.1 Fuzzy Sets

A classical set is a set with a crisp boundary. The transition from “belonging to a set” to “not belonging to a set” is crisp. If an element belongs to the set, its membership is 1. If an element does not belong to the set, its membership is 0.

The main concept of fuzzy theory is fuzzy set. A fuzzy set is a set of that does not have a crisp boundary. It is characterized by a membership function. A fuzzy set is denoted by an ordered set of pairs, whose element is denoted as x :

$$A = \{(x, \mu_A(x)) | x \in X\} \quad (7.1)$$

where μ_A takes values in the interval $[0, 1]$. If x does not belong to the fuzzy set A , the degree of membership function is 0. If x belongs to the fuzzy set A , the degree of membership function changes in the interval $[0, 1]$.

7.8.2 Membership Function

The meaning of the linguistic variables is quantified by using membership functions. A membership function is a curve that assigns to each point in the input space is mapped to a membership value which is discrete value defined in the $[0,1]$. There are a lot of shapes of a membership function depends on application. In Figure 1, the membership functions most commonly used in control theory are monotonic, triangular, trapezoidal or Gaussian.

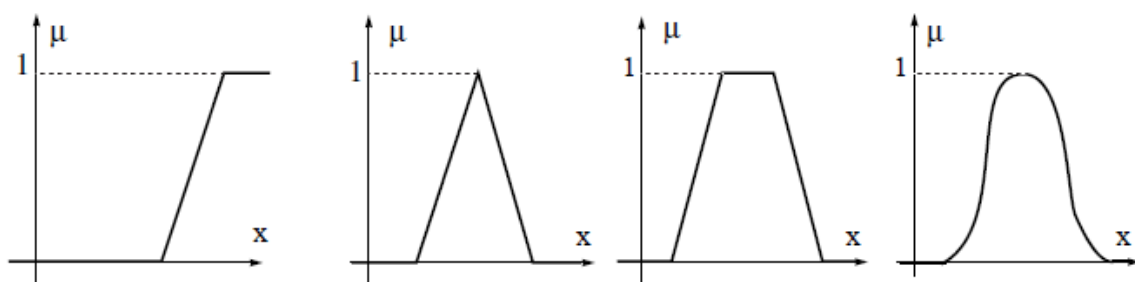


Figure 7.16: Monotonic, triangular, trapezoidal and bell-shaped membership functions

7.8.3 Operations with Fuzzy Sets

A and B are two fuzzy sets with membership functions μ_A and μ_B respectively.

7.8.3.1 Union Operation

Union of the two fuzzy sets is defined as a lot of shape. Max operator, algebraic sum and bounded sum are some of them. Max operator chooses the maximum of the two individual membership functions.

$$\mu_{A \cup B} = \max(\mu_A, \mu_B) \quad (7.2)$$

7.8.3.2 Complement Operation

Complement of a fuzzy set is defined as the negation of the membership function.

$$\mu_{\hat{A}} = 1 - \mu_A \quad (7.3)$$

7.8.3.3 Intersection Operation

Intersection of the two fuzzy sets can be defined by using min operator, algebraic product or bounded product. Min operator chooses the minimum of the two individual membership functions.

$$\mu_{A \cap B} = \min(\mu_A, \mu_B) \quad (7.4)$$

7.8.4 Fuzzy Control Structure

General structure of fuzzy inference system is shown in Figure 2. The fuzzy controller is composed of the following four blocks:

1. A rule-base contains a fuzzy logic quantification of the expert's linguistic description of how to achieve good control. The general form of linguistic rules is:

IF *premise proposition* **THEN** *statement*

2. An inference mechanism determine which rules are relevant to the current situation
3. A fuzzification interface calculates the membership function values for an input variable and use this information by using the rule-base table.

4. A defuzzification interface determines a crisp value that best represents the fuzzy set. There are a number of defuzzification methods provides a means to choose a single output such as center of area (COA), mean of maximum (MOM), largest of maximum (LOM) and smallest of maximum (SOM).

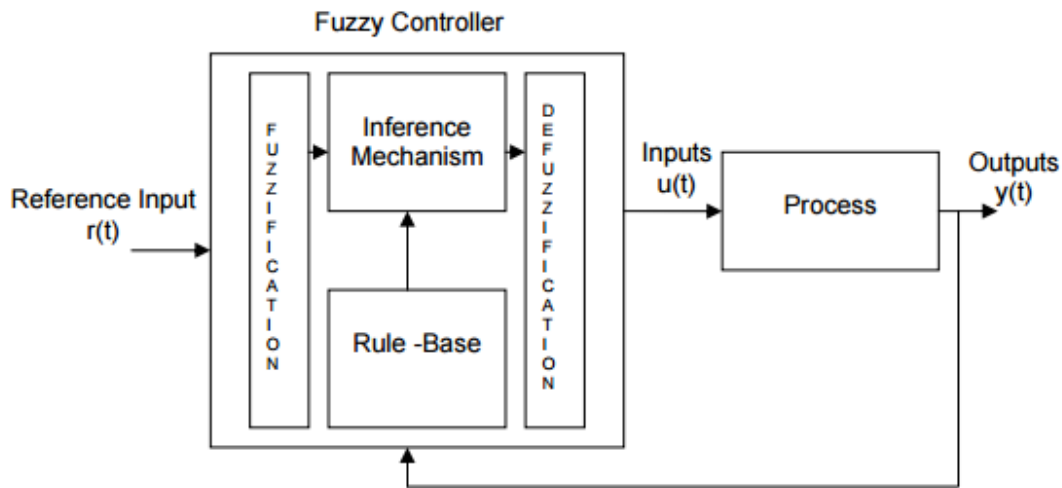


Figure 7.17: Fuzzy control structure

7.8.5 Fuzzy Rule Structures

7.8.5.1 Mamdani Type Fuzzy Rule Base

In the fuzzy rule base, states and statements are defined by fuzzy.

$$\text{IF } \tilde{x} A_i \text{ THEN } \tilde{y} B_i$$

\tilde{x} is the linguistic variable and A_i is input membership function. In the same way, \tilde{y} is the linguistic variable and B_i is output membership function. In this here, all linguistic variables and linguistic terms are defined in their region and membership functions are fuzzy sets which are defined between 0 and 1.

7.8.5.2 Takagi – Sugeno Type Fuzzy Rule Base

The output membership functions are defined by linear functions that are different from Mamdani rule base in Takagi-Sugeno type fuzzy rule base.

$$\text{IF } \tilde{x} A_i \text{ THEN } y_i = f_i(x), \quad i = 1, 2, \dots, n$$

7.8.5.3 Single Type (Singleton) Fuzzy Rule Base

This rule base, which is named as originally Singleton, is specified version of Takagi - Sugeno Type Fuzzy Rule Base. The output membership functions are defined by constant values.

$$\text{IF } \tilde{x} A_i \text{ THEN } y_i = b_i, \quad i = 1, 2, \dots, n$$

7.8.6 Design of the Fuzzy Logic Controller

When a fuzzy controller is designed, informations which will be used as inputs are determined at first. In the ball and beam system, the fuzzy controller has two inputs and one output, namely position error, velocity and angle respectively. Position error input is defined as the difference between desired position and actual position. Velocity input is defined as derivative of position error. After, the fuzzy inputs are determined, the input and output membership functions of fuzzy controller are determined.

One of the factors which cause to a change on the controller performance is the number of fuzzy sets in the membership functions. Systems which have fast responses such as ball and beam, ball and plate, 1-Dof helicopter generally have five or more fuzzy sets in their membership functions. Other factor which causes to a change on the controller performance is shape of membership functions. Triangular membership function is commonly used for the systems. Also, Gaussian membership function can be used in order to increase the nonlinearity of the fuzzy controller.

7.8.6.1 Creating Membership Functions of Fuzzy Logic Controller

First of all, Fuzzy System Designer should be open by selecting “*Tools>>Control Design and Simulation>>Fuzzy System Designer*”. In the popup window, input variables and output variables can be added by clicking the “*Add Input Variable*” and “*Add Output Variable*”.

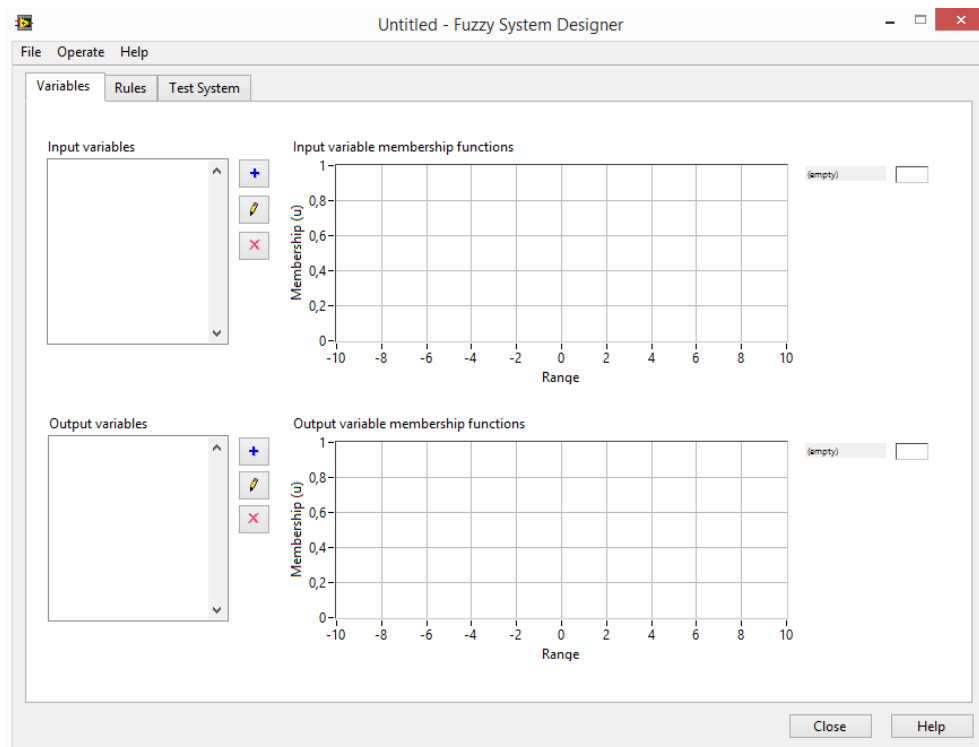


Figure 7.18: Fuzzy system designer - Variables

After an input or an output variable is added, “*Edit Variable*” dialog box is opened. Range, shape and number of membership functions can be adjusted by using the box. A membership function is created by clicking the “*Add Membership Function*”.

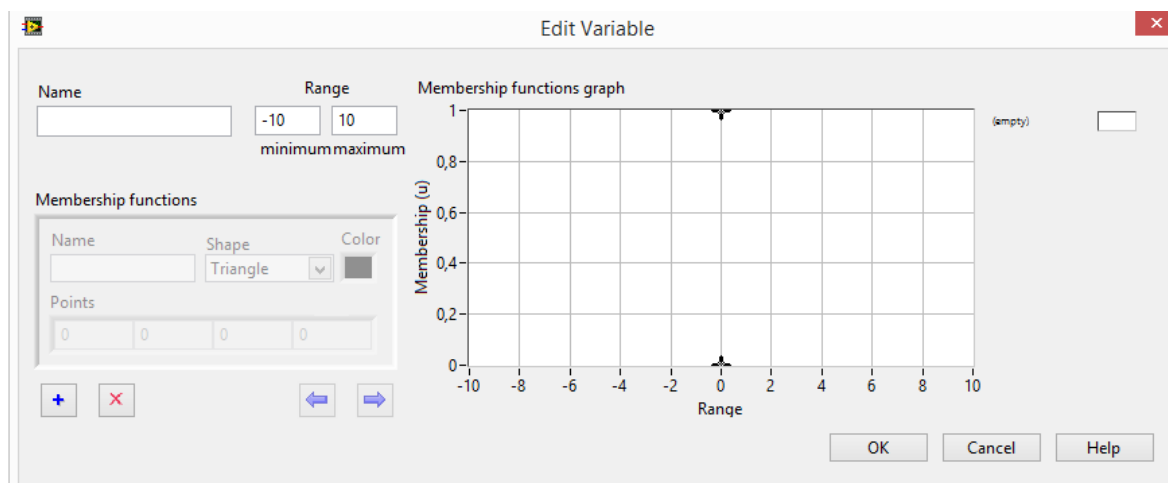


Figure 7.19: Edit variable dialog box

The range of inputs and output is determined in $[-1, 1]$. The number of membership functions is chosen as 5 both of the inputs and inputs membership functions are triangular membership function. The rule - base table for fuzzy controller is shown in Table 7.2[1]. In this

study, single type (Singleton) rule base is used due to the simplicity of defuzzification section, so the shape of output membership function is chosen as singleton. Finally, the inputs and output membership functions are shown in Figure 7.20.

e \ w	NH	NL	Z	PL	PH
NH	-1	-0.7	-0.5	-0.3	0
NL	-0.7	-0.4	-0.2	0	0.3
Z	-0.5	-0.2	0	0.2	0.5
PL	-0.3	0	0.2	0.4	0.7
PH	0	0.3	0.5	0.7	1

Table 7.2: Fuzzy rule base

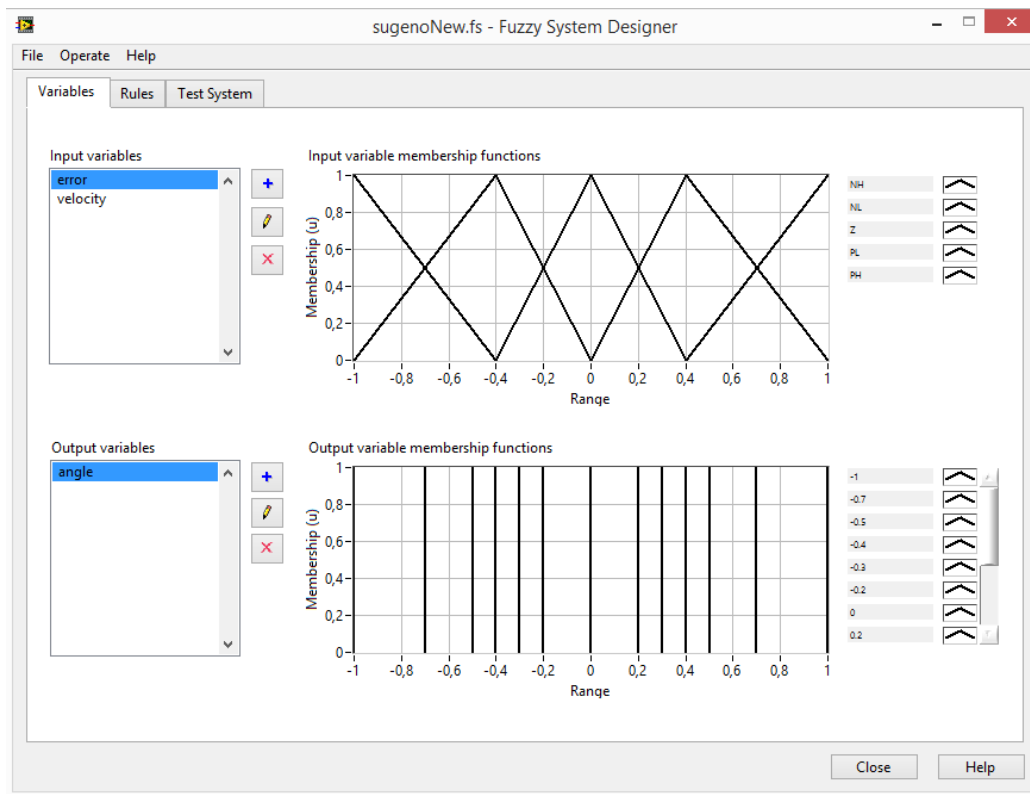


Figure 7.20: Fuzzy System Designer – Input and Output membership functions

After inputs and output membership functions are determined, the rule table of the fuzzy controller is obtained. A new window is opened by clicking the “Rules” tab of the Fuzzy System Designer (Figure 7.21). The rules can be added by clicking the “Add Rule” button to the right of the “Rules” list. Also, Defuzzification method which provides to convert the degrees of membership function into a single output can be chosen in the window.

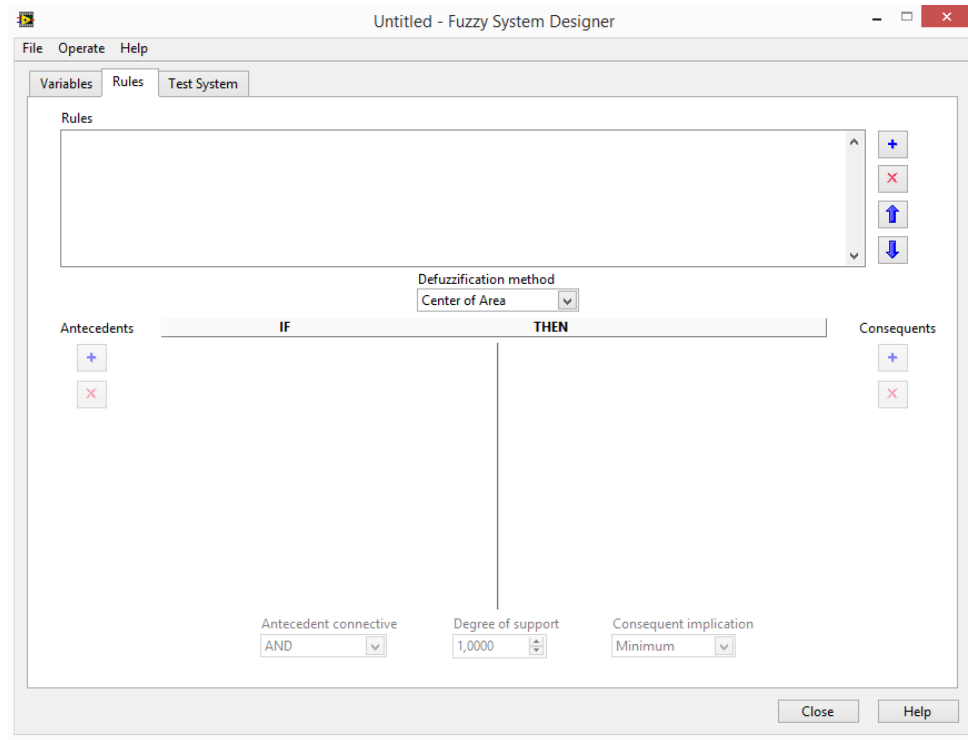


Figure 7.21: Fuzzy System Designer – Rules

After the fuzzy controller is designed, the fuzzy structure is determined. In the ball and beam system, a fuzzy PID structure is used in order to achieve a good tracking performance. The fuzzy PID structure is shown in Figure 7.22.

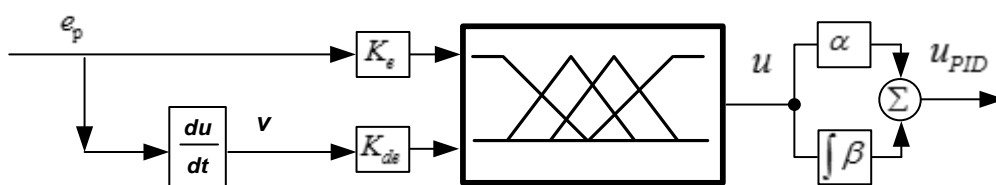


Figure 7.22: Fuzzy PID structure

K_e and K_{de} are scaling factors of the fuzzy PID controller. They are used in order to set the error and velocity inputs in the interval of input membership functions which are in $[-1, 1]$.

In the ball and beam system, the input universe of discourse for position error is $[-25, 25]$, so K_e should be $1/25=0.04$ [2]. Scaling factor of velocity input (K_{de}) should be chosen as the value of K_e (0.04) or it can be found experimentally in simulation. The determination of the scaling factor of fuzzy outputs is done by using similar way. In order to find α coefficient, input of system is investigated. The input of the ball and beam system is angle of motors and the output universe of discourse for angle of motors is $[-45, 45]$. Hence, α can be chosen as 45. However, control signal (angle of motor) change in $[-22.5, 22.5]$ when α is 45. Hence, α is chosen as 60 in order to make the ball and beam system faster. If α is chosen higher than 60, control signal is getting worse. A lot of experiments are done in simulation in order to find integrator coefficient (β) and it is found as 7.

There are some ways to find out the scaling factors of the fuzzy controller in literature. However, in this work, scaling factors of the fuzzy controller are found as explained above.

After the scaling factors of the fuzzy controller are found, the step responses of the model and real system are shown in Figure 7.23.

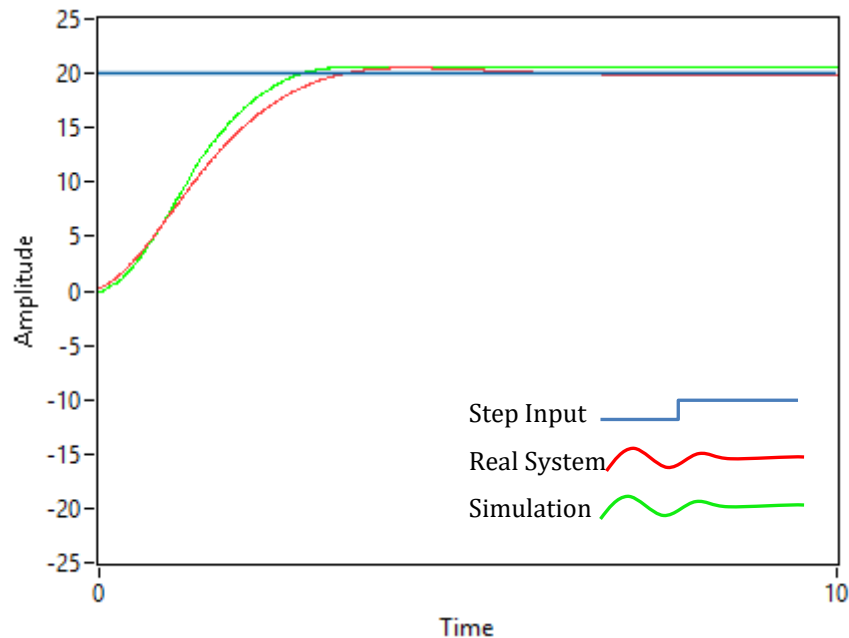


Figure 7.23: The Real-Time, Simulation and Reference graphs with Fuzzy Controller

7.8.6.2 In – Lab Exercise

1. Design a fuzzy controller via Fuzzy System Designer in Labview. Use the Fuzzy rule-table below and use prototype membership function is shown in Figure 7.24 for all inputs (position error and angular velocity) and output (angle).

$\begin{matrix} e \\ w \end{matrix}$	N	Z	P
N	N	N	Z
Z	N	Z	P
P	Z	P	P

Table 7.2: Fuzzy rule base

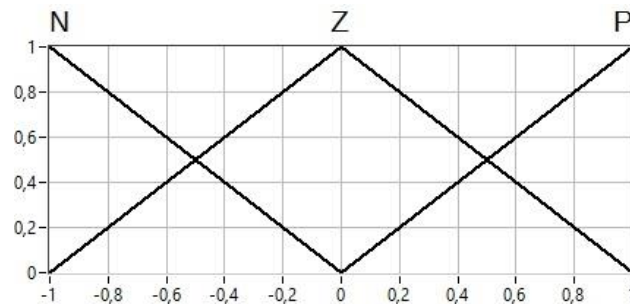
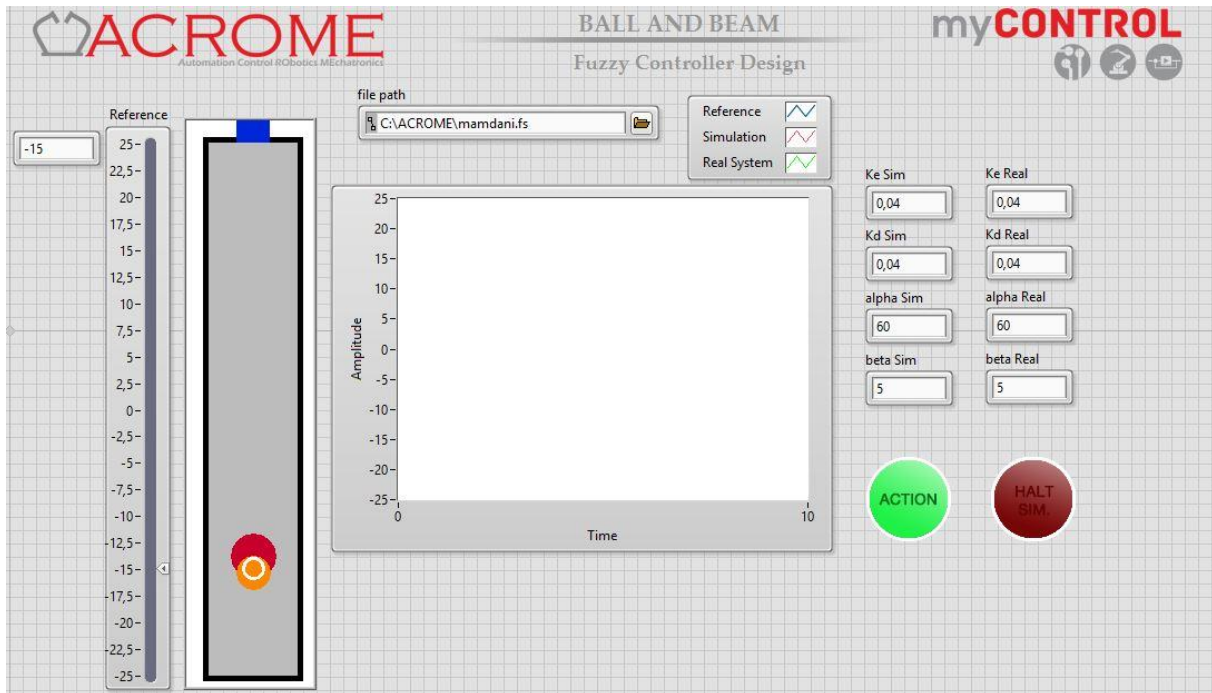


Figure 7.24: Inputs and Output membership functions

2. Save the designed fuzzy controller.
3. Use fuzzy PID structure in order to control ball and beam system.
4. Find the scaling factors of fuzzy PID controller as explained above.
5. Investigate the system response for scaling factors which you obtain. If the system response is unstable or not good, use 0.04, 0.04, 60 and 5 for scaling factors K_e , K_{de} , α and β respectively.
6. Compare the system responses of 5x5 fuzzy controller and 3x3 fuzzy controller in terms of settling time and overshoot. Which controller has better reference tracking performance?

7.8.6.3 In – Lab Exercise

1. Open “Fuzzy Control.vi” as follows,



2. Select “sugenoNew.fs” file.
3. Enter the values 0.04, 0.04, 60 and 5 for scaling factors K_e , K_{de} , α and β respectively and investigate the system response.
4. Change K_{de} , α and β coefficients one by one. Interpret effects of change in K_{de} , α and β on system response.

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.