Design and Development of a test rig for testing of Fuzzy controller

Submitted by:
Danish Karim Iqbal

Supervised by:
Dr. Muhammad Bilal Kadri

THESIS
Submitted to:
Department of Electronic and Power Engineering
Pakistan Navy Engineering College Karachi
National University of Science and Technology, Islamabad Pakistan

In fulfillment of requirement for the award of the degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING
With Specialization in Control Engineering

بسم الله الرحمن الرحيم

ABSTRACT

Increasing product quality and production rate of industries at the same time keeping down the cost is the main issue, process control techniques deal with today. In process control, precise level control of liquid in reaction vessels and storage tanks is essential. This work attempts at designing, testing, and developing a coupled-tank model for testing different control schemes. It starts with introducing coupled tank systems and then move on to developing a mathematical model for it. In the next section complete construction of tank is discussed, problems faced, how we overcame them, and how the tank was interfaced with MATLAB Simulink software. In the end two different control schemes are tested out on the tank and their results published: PID controller and adaptive fuzzy controller.

ACKNOWLEDGEMENT

First and foremost, I am obliged to thank Allah Subhana-watala for giving me all the blessings of the world in terms of my family, in terms of health, in terms of opportunities, in terms of the required ability, in terms of the intellect to think and learn, in terms of financial support and in terms of my existence as a whole. If all this was not bestowed upon me, I would never be able to do anything in life. All Praise to Allah Subhana-watala who gave me a supportive advisor like Dr. Muhammad Bilal Kadri. His relentless support during my entire work is commendable beyond words. I would finally thank my GEC members for their help and sincere wishes.

Table of Contents

List of Figures

LIST OF TABLES

ABBREVIATIONS

2-DOF	2 Degrees of Freedom
AC	Alternating Current
ADC	Analog to Digital converter
CTS	Coupled Tanks System
DC	Direct Current
FMIGO	Fractional M_8 Constrained Integral Gain Optimization
FO-PI	Fractional Order Proportional Integral Controller
LSB	Least Significant Bit
MIMO	Multi-Input/Multi Output
MPC	Model Predictive Control
MRAC	Model Reference Adaptive Control
MZN	Modified Ziegler-Nichols
PID	Proportional-Integral-Derivative
SISO	Single-Input/Single-Output
SMC	Sliding Mode Control
TITO	Two-Input/Two-Output
UART	Universal Asynchronous Receiver Transmitter
ZN	Ziegler-Nichols

1 INTRODUCTION

1.1. Motivation

Increasing product quality and production rate of industries at the same time keeping down the cost is the main issue, process control techniques deal with today. In process control, precise level control of liquid in reaction vessels and storage tanks is essential especially in chemical engineering industry where chemicals are pumped from one tank to another for storage or pumping to another tank, all this is done using coupled tanks[1]. Essentially, coupled tank systems has applications in many industrial and commercial sectors. Coupled tanks are essential part of the plant in many industrial applications such as beverage industry[2], food processing, filtration, industrial chemical processing, water purification system, spray coating[3] and pharmaceutical industries[4]. A typical situation can be where the chemical reactor is fed with a liquid at a constant rate. Two tanks can be used where the upper tank performs filtering of variations in the upstream supply flow and at the lower tank the level of the liquid remains constant to give a constant flow. The liquid will be circulated many times as well as it will be processed chemically mixed, but the level of the liquid is to be kept constant through out the process.

Most of the systems found in these industries are multi-input multi output (MIMO) systems. When designing a controller for these systems it helps to note that these systems are inherently nonlinear making the tuning of the controller further difficult[5].

It is therefore crucial for the control engineer to understand how liquid level is controlled and how coupled tank system work. System dynamics and interacting characteristics contribute to the problem of level control. In order to understand these dynamics often mathematical models are available to control engineers. Although every effort is made to accurately describe the system using a systems approach, a model still would not include every aspect of the situation. A model is always an abstraction that is simpler than the original plant. Only the aspects which are related to the problem is accounted for in the hope that whatever is of value, with regard to the original problem, is covered[6]. In order for a young control engineer to study these systems practically is very difficult and sometimes it is not be possible to intervene a running system to test a controller. A testing rig which can substitute actual tank, where it would not incur loss to the owner, if the controller does not perform as expected and where the controller can be tested a hundred times before deploying on the original plant, becomes essential. Many companies have products which mimic the actual plant but they are too expensive for the universities of third world countries. A cheaper alternative was necessary in order for students to learn the practical implications of deploying a controller.

1.2. Objectives of thesis

The primary objective of this thesis is to design and develop a testing rig, which is a coupled tank system, to test controllers built on MATLAB or Simulink environment. Following are the sub-objectives taken together from the main objective.

- In depth study of coupled-tank systems.
- Mathematical modeling of coupled tank system.
- Implementation of a PID controller on coupled tank system.
- Implementation of an adaptive fuzzy controller on coupled tank system.

1.3. Thesis organization

The thesis has been organized into six chapters.

Chapter 1: Gives the introduction behind the design and development of a coupled tank system, objective of the thesis and thesis organization.

Chapter 2: This chapter includes the major relevant research work studied in depth to help in this thesis work.

Chapter 3: In this chapter coupled tank system is discussed. Its mathematical model is prepared and open loop responses are also shown.

Chapter 4: In this chapter the construction of tank is discussed, problems faced and how they were countered and open loop response of the tank is also shown in single-input single-output(SISO) and multi-input and multi-output(MIMO) configurations.

Chapter 5: This chapter discusses implementation of PID controller and adaptive fuzzy logic controller. PID controller gains are calculated using Ziegler-Nichols rules for tuning PID. Response of the tank is shown due to a continuously changing set point. Then the tank is tested with an adaptive fuzzy controller which tries to maintain the level at varying set-points.

Chapter 6: This chapter presents conclusion and future work recommendation.

2 LITERATURE REVIEW

2.1. Introduction

The chapter presents a literature review of the studies carried out related to coupled tank system and the approaches used to maintain the level of liquid. For the sake of compactness, all of the corresponding material is not brought into the scope of the chapter however some key references will be discussed briefly. A detail list of the reference material is provided in reference section for further study.

2.2. Major research studies

2.2.1 Development of a Web-Based Laboratory for Control Experiments on a Coupled Tank Apparatus[7]

This paper aimed at making a web based laboratory for easy access to any one any where wherever there is Internet. This concept is not new; Shor and Bhandari[8] developed an application that allowed users to remotely conduct experiments in the Control Engineering Laboratory at Oregon State University. Via World Wide Web the application permitted the experiments to be run on actual hardware remotely. On the client side only a browser which ran Java was used to gain access to the remote hardware. The hardware used in this paper is setup at National University of Singapore where a coupled tank apparatus is setup for web based control. The researchers have implemented the capability to remotely apply manual, PID, fuzzy logic control and general state-space control strategies to the coupled tank equipment. A video conferencing is also setup with camera mounted on a movable platform which can be controlled by user. Both zooming and viewing angle is remotely configurable.

The lab was hosted at http://vlab.ee.nus.edu.sg/vlab/control but is not accessible anymore.

2.2.2 Remote Experimentation on a Three Coupled Water Reservoirs [9]

This paper discusses remote laboratory similar to the web based laboratory for experimenting with coupled tank system. The researchers have made a web application using Java, which allows user to control and visualize the experiment in real-time. User can choose from 4 options to implement control system which are: PID, state-space, Fuzzy and Exact Linearization.

User opens the link and a Java applet is installed and then he/she can implement any control strategy they want. Real time streaming was done using Real Media player.

2.2.3 Robust PID controller design for coupled-tank process [10]

In this paper the researchers design a robust PID controller for coupled tank system in the frequency domain. The coupled tank system is configured in SISO mode and only one tank is used, no coupling. Two approaches have been used to design the robust controller: Edge theorem and modified Neimark's D-partition which ensures phase margin.

Results for both the schemes are illustrated at the end of the paper which suggests that the modified Neimark's D-partition which ensure phase margin has better control quality than the Edge theorem technique.

2.2.4 Experimental Study of Fractional Order Proportional Integral (FOPI) Controller for Water Level Control [11]

This paper studies coupled tank systems and compares different controllers to control the level of liquid in the tank. This paper is written in four steps; mathematical modeling of the plant, identification of plant parameters, controller design and comparisons, first in simulation then on actual hardware.

Controller design in this paper is based on Fractional Order Proportional Integral Controller (FO-PI). This is then compared to Ziegler Nichols(ZN) and Modified Ziegler-Nichols (MZN). Comparison is based on load disturbance rejection, set-point tracking and changes in plant dynamics.

Researchers used PP-100 tank in three different configurations. For each configuration of tank three techniques were used to tune the PI gains: Ziegler Nichols(ZN), Modified Ziegler Nichols and Fractional M_8 Constrained

Integral Gain Optimization (FMIGO) . In all the three cases simulation results agreed well with the actual hardware run even though there were real disturbances due to ware dynamics, non- linearities in the system and modeling uncertainties. The results concluded that FO-PI tuned by FMIGO performs better than others and can be a promising controller for coupled tank systems.

2.2.5 Sliding Mode Control, with Integrator, for a Class of MIMO nonlinear Systems[12]

This paper proposes the robust control problem of general nonlinear multi-input multi-output (MIMO). The robustness against unknown disturbances is considered. Two algorithms based on the Sliding Mode Control (SMC) for nonlinear coupled Multi-Input Multi-Output (MIMO) systems are proposed: the first order sliding mode control (FOSMC) with saturation (sat) function and the FOSMC with sat combined with integrator controller.

These algorithms were tested on simulation and then on actual coupled tank system and their results published.

With the help of saturation and integral controller chattering in the controller output was visibly less and the steady state error was nearly zero.

The researchers concluded this paper by publishing results of simulation along with real system testing. They showed results with varying cross-section of coupling valves and leakage valves to show the stability of the controller and rejection of disturbance.

2.2.6 Design of 2-DOF PI Controller with Decoupling for Coupled-Tank Process[13]

In this paper, an approach to design the 2 Degree Of Freedom(DOF) Proportional Integral (PI) controller with decoupling for a coupled tank system in discussed. The decoupling control transforms the MIMO system to two SISO system which is then controlled by a 2-DOF PI controller. The coupled tank system used is a two tank coupled system with two input and two output.

At first the researchers tested the 2-DOF PI controller without the decoupling and then with decoupling. The results show that decoupling reduces the overshoot and settles quickly than without decoupling. Overshoot with decoupling is less than 5% and settling time is reduced to 40seconds.

3 DETAILED DISCUSSION OF MATHEMATICAL MODEL OF MIMO AND SISO COUPLED TANK SYSTEMS

3.1. Introduction:

In this chapter the nonlinear plant (coupled tanks system) will be discussed. The linearized state space model of the plant and the model parameters will determined. In this chapter, the mathematical modeling of the plant will be performed using first principle method.

3.2. Overview of coupled tank system

In the process industries the control of liquid level in tanks and regulated flow between tanks is a basic problem. In order to learn the behavior of the coupled tank system a model is developed. The coupled tanks system consists of three vertical tanks interconnected by a flow channel as shown in Figure 3.1. Left and right tank has an independent pump for inflow of liquid in to the tank. The cross-sectional area of the outlets present at the base of each tank and the channel connecting the left and middle tanks or right and middle tanks can be varied manually. Tank structures similar to coupled tank system model presented here are essential in various industries such as, petrochemical, paper making and water treatment industries. The unique thing about coupled tanks system is that, since the tanks are coupled together the levels interact and must be controlled. The coupled tanks system can be configured as a Single Input Single Output (SISO) or as a Multi Input Multi Output system(MIMO).

3.3. Determination of Model parameters:

The system model used here can be determined by various methods. There are two ways of developing the mathematical of the coupled tanks system:

- By performing a series of tests on the plant, signals such as step, sine, pseudo-random are applied to the system and the output is saved. Then a system identification scheme is employed to obtain the linear plant model.
- By using the First Principles method. In this method the plant dynamics can be described by using nonlinear equations. For the derivation of the nonlinear equations the complete knowledge of the process in the system such as thermodynamic, chemical processes and the physical specification is required.

For this thesis the First Principles method is used to obtain the nonlinear process model.

3.3.1 Mathematical Modeling of MIMO Coupled Tanks System Using First Principle Approach

The first principle method uses application of conservation of mass known as mass balance.

The plant considered here is a three tank system. It has three outputs and two inputs. It consists of a rectangular acrylic tank separated into three sections so as to give three tanks. Each section a_1, a_2 and a_3 has the dimensions as shown in table 3.1:

Section	Width(mm)	Height(mm)
a_1	90	104
a_2	90	104
a_3	90	104

Table 3.1: Base dimensions of each tank

The pipe connecting Tank 1 and Tank 3 have a manually adjustable ball valve a_{z1}. Similarly the pipe connecting Tank 2 and Tank 3 also have a manually adjustable ball valve a_{z2}. There are also three more pipes each connected with their respective tanks and each have a manually adjustable ball valve, b_{z1}, b_{z2} and b_{z3}, and lead to the reservoir tank. There are two pumps, pump 1 and pump 2, which are connected to the reservoir tank to provide the water to tank 1 and tank 2. Their flow rates are $Q1(t)$ and $Q2(t)$ respectively. The height measurements, $h_1(t)$, $h_2(t)$ and $h_3(t)$, are carried out by piezo-resistive differential pressure transducers. To obtain the state equations we equate the variation of water volume in a tank with the difference between the incoming flows and the outgoing flows, this means that the flow toward the tank 3 can come from tank 1 and tank 2.

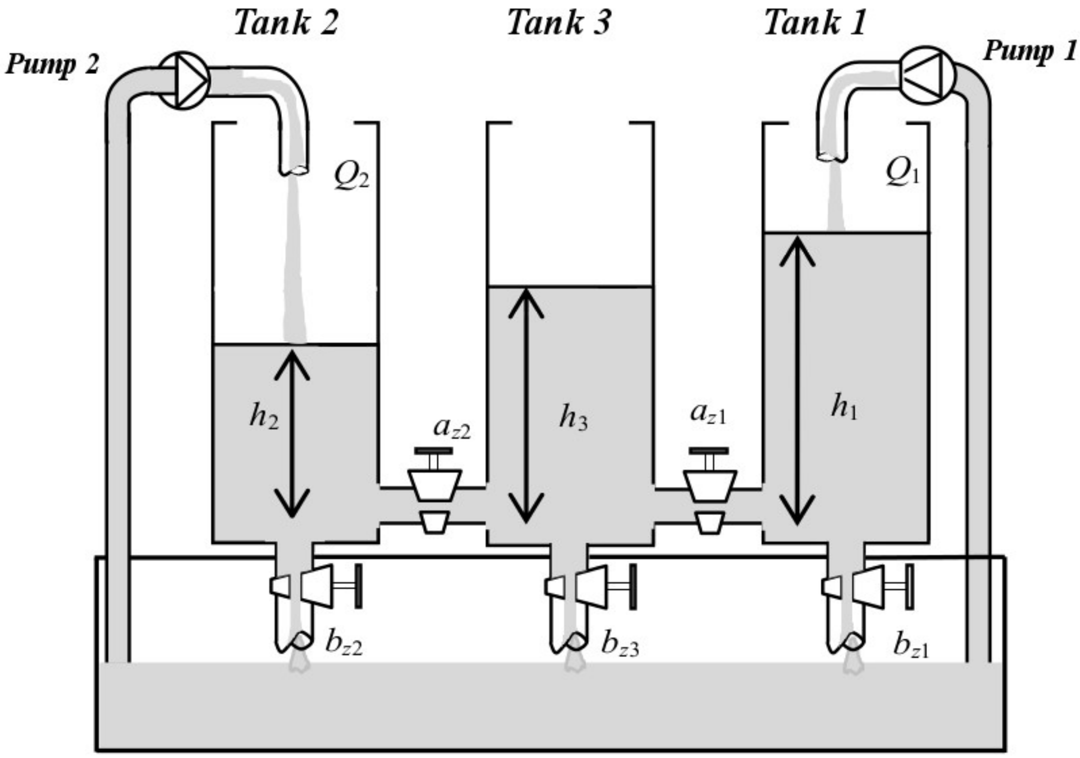

Figure 3.1: Layout of coupled tank system.

Then the system can be represented by the following equation:

$$\dot{h}_m(t) = \frac{1}{A}\left(Q_m^{\text{in}}(t) - Q_{mn}^{out1}(t) - Q_{mn}^{out2}(t)\right) m, n = 1,2,3 \tag{1}$$

Where:

$Q_m^{\text{in}}(t)$ is the flow through pump *m* (*m* = 1,2) and $Q_m^{\text{out1}}(t)$ represents the flow rates of water between the tanks *m* and *n* (*m, n* = 1,2,3 where *m* ≠ *n*). It is expressed using Torricelli's law

$$Q_{mn}^{\text{out1}}(t) = a_{zm} S_n \, sign(h_m - h_n) \sqrt{2\text{g}|h_m - h_n|} \quad m, n = 1,3 \tag{2}$$

and $Q_m^{\text{out2}}(t)$ represents the outflow rate, given by:

$$Q_{mn}^{\text{out2}}(t) = b_{zn} S_L \sqrt{2\text{gh}_n} \quad n = 1,2,3 \tag{3}$$

The parameters of this tank is defined in table 3.2.

Symbol	Value	Meaning
a	0.00936 m^2	Tank 1 base area
S_n	1.266 x 10^{-4} m^2	Valve cross-section area
a_{zm}	$0 \leq a_{zm} \leq 1$	Flow correction term (m=1,2)
b_{zm}	$0 \leq b_{zm} \leq 1$	Leakage flow correction term (m = 1,2,3)
g	9.81 m/s^2	Acceleration due to gravity
h_{max}	0.6 m	Maximum water level in tank (m=1,2,3)
Q_{1max}	0.0006 m^3/s	Maximum flow through pump 1
Q_{2max}	0.00028 m^3/s	Maximum flow through pump 2

Table 3.2: Physical parameters of the tank

Now the controlled signals are the water levels in each tank (h_1, h_2 and h_3) of tank 1, tank 2 and tank 3 respectively. These levels, as already described, are controlled by two pumps. To consider a MIMO system we choose two tank's height since all three cannot be controlled to give a specific height but two or any one of them can be controlled. We take two tanks for our MIMO plant model.

3.3.2 MIMO plant model

As the water level can be controlled in only two of the tanks we chose tank 2 (right tank) and tank 3 (middle tank) for liquid level control which gives us h_2 and h_3 parameters respectively, as shown in the figure 3.2. Now the three tanks can be modeled by the three differential equation as shown in (4) (5) and (6).

$$\frac{dh_1}{dt} = -c_1\, sign(h_1 - h_3)\sqrt{|h_1 - h_3|} - B_1\sqrt{h_1} + \frac{Q_1}{a} \tag{4}$$

$$\frac{dh_2}{dt} = c_3\, sign(h_3 - h_2)\sqrt{|h_3 - h_2|} - B_2\sqrt{h_2} + \frac{Q_2}{a} \tag{5}$$

$$\frac{dh_3}{dt} = c_1\, sign(h_1 - h_3)\sqrt{|h_1 - h_3|} - B_3\sqrt{h_3} - c_3\, sign(h_3 - h_2)\sqrt{|h_3 - h_2|} \tag{6}$$

Where c_n, n = 1, 3 and B_n, n = 1,2,3 are as follows :

$$c_n = \frac{1}{a} a_{zi} S_n \sqrt{2g} \qquad n = 1,3 \tag{7}$$

$$B_n = \frac{1}{a} b_{zn} S_L \sqrt{2g} \qquad n = 1,2,3 \tag{8}$$

The parameters a, a_{zi}, b_{zn}, S and g are defined in table 3.2

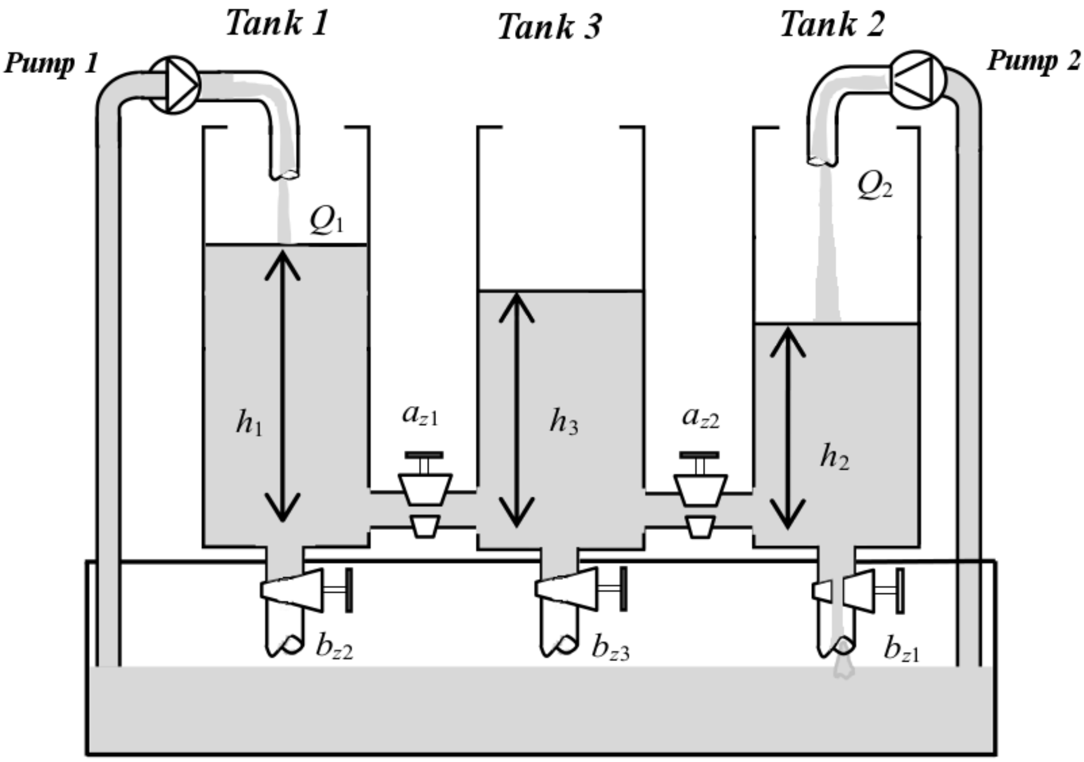

Figure 3.2: Tank configured as MIMO plant with ball valve b_{z2} and b_{z3} fully closed and b_{z1} partially open. Control variables are h_3 and h_2.

While tanking $B_1 = B_3 = 0$, as only the tank 2 outlet valve is open rest are all closed, the three equations become:

$$\frac{dh_1}{dt} = -c_1\, sign(h_1 - h_3)\sqrt{|h_1 - h_3|} + \frac{Q_1}{a} \tag{9}$$

$$\frac{dh_2}{dt} = c_3\, sign(h_3 - h_2)\sqrt{|h_3 - h_2|} - B_2\sqrt{h_2} + \frac{Q_2}{a} \tag{10}$$

$$\frac{dh_3}{dt} = c_1\, sign(h_1 - h_3)\sqrt{|h_1 - h_3|} - c_3\, sign(h_3 - h_2)\sqrt{|h_3 - h_2|} \tag{11}$$

When the water level becomes constant we reach equilibrium, hence the derivatives become zero.

$$\dot{h}_1 = \dot{h}_2 = \dot{h}_3 = 0 \tag{12}$$

Using (12) in the steady state we get the following algebraic relationship

$$-c_1\, sign(h_1-h_3)\sqrt{|h_1-h_3|}+\frac{Q_1}{a}=0 \tag{13}$$

$$c_3\, sign(h_3-h_2)\sqrt{|h_3-h_2|}-B_2\sqrt{h_2}+\frac{Q_2}{a}=0 \tag{14}$$

$$c_1\, sign(h_1-h_3)\sqrt{|h_1-h_3|}-c_3\, sign(h_3-h_2)\sqrt{|h_3-h_2|}=0 \tag{15}$$

In this tank the pumps can only pump in the liquid and not drain it therefore Q_1 and Q_2 cannot be negative:

$$\begin{gathered} Q_1\geq 0 \\ Q_2\geq 0 \end{gathered} \tag{16}$$

From (13) and (15) we have

$$c_1\, sign(h_1-h_3)\sqrt{|h_1-h_3|}=\frac{Q_1}{a} \tag{17}$$

$$c_1\, sign(h_1-h_3)\sqrt{|h_1-h_3|}-c_3\, sign(h_3-h_2)\sqrt{|h_3-h_2|} \tag{18}$$

Then $(h_3 - h_2) \geq 0$ and $(h_1 - h_3) \geq 0$. If we assume.

$$x_1=h_1,\ x_2=h_2,\ x_3=h_{3,}\ u_1=Q_1 \text{ and } u_2=Q_2 \tag{19}$$

We have:

$$\begin{gathered} \dot{x}_1=-c_1\sqrt{|x_1-x_3|}+\frac{u_1}{a} \\ \dot{x}_2=c_3\sqrt{|x_3-x_2|}-B_2\sqrt{x_2}+\frac{u_2}{a} \\ \dot{x}_3=c_1\sqrt{|x_1-x_3|}-c_3\sqrt{|x_3-x_2|} \end{gathered} \tag{20}$$

These state equations (20) can now be written in the form of (21)

$$\begin{gathered} \dot{x}=f(x,t)+gu \\ y=cx \end{gathered} \tag{21}$$

Where

$$x=[x_1 \quad x_2 \quad x_3]^T, u=[u_1 \quad u_2]^T, y=[x_2 \quad x_3]^T$$

$$f(x,t)=\begin{pmatrix} -c_1\sqrt{x_1-x_3} \\ c_3\sqrt{x_3-x_2}-B_2\sqrt{x_2} \\ c_1\sqrt{x_1-x_3}-c_3\sqrt{x_3-x_2} \end{pmatrix}$$

$$g=\begin{pmatrix} \frac{1}{a} & 0 \\ 0 & \frac{1}{a} \\ 0 & 0 \end{pmatrix} \quad \text{and} \tag{22}$$

$$c=\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3.3.3 SISO plant model

To have a SISO model we can turn off pump 2 and control the liquid level in tank 3 leave the rest of the configuration as it is as shown in figure 3.3.

Now we have

$$u_1=Q_1 \text{ and } u_2=0 \tag{23}$$

Using (23), (20) now becomes :

$$\begin{aligned} \dot{x}_1 &= -c_1\sqrt{|x_1-x_3|}+\frac{u_1}{a} \\ \dot{x}_2 &= c_3\sqrt{|x_3-x_2|}-B_2\sqrt{x_2} \\ \dot{x}_3 &= c_1\sqrt{|x_1-x_3|}-c_3\sqrt{|x_3-x_2|} \end{aligned} \tag{24}$$

(21) Now becomes :

$$\dot{x} = f(x,t) + gu \qquad (25)$$
$$y = cx$$

Where

$$x = [x_1 \quad x_2 \quad x_3]^T, u = u_1, y = x_3$$

$$f(x,t) = \begin{pmatrix} -c_1\sqrt{x_1 - x_3} \\ c_3\sqrt{x_3 - x_2} - B_2\sqrt{x_2} \\ c_1\sqrt{x_1 - x_3} - c_3\sqrt{x_3 - x_2} \end{pmatrix} \qquad (26)$$

$$g = \begin{pmatrix} \frac{1}{a} \\ 0 \\ 0 \end{pmatrix} \quad \text{and}$$

$$c = (0 \quad 0 \quad 1)$$

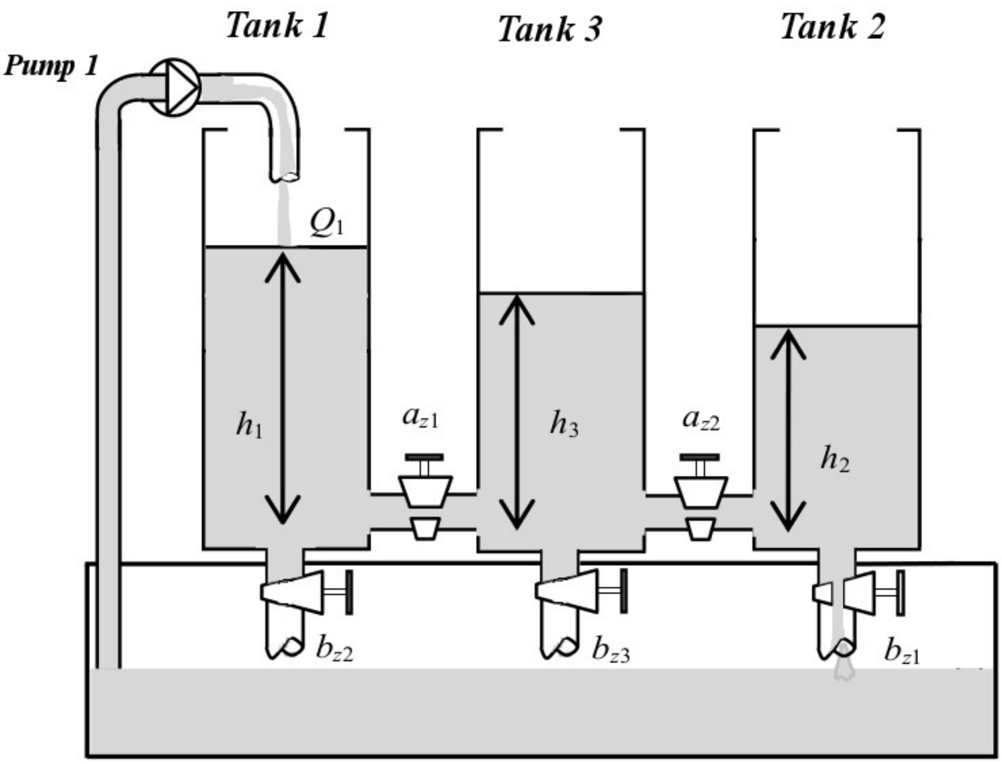

Figure 3.3: Tank configuration for SISO plant with ball valve b_{z2} and b_{z3} fully closed and b_{z1} partially open and pump 2 removed. Control variable is h_3.

3.3.4 Response of SISO and MIMO

3.3.4.1 Plant model in Simulink

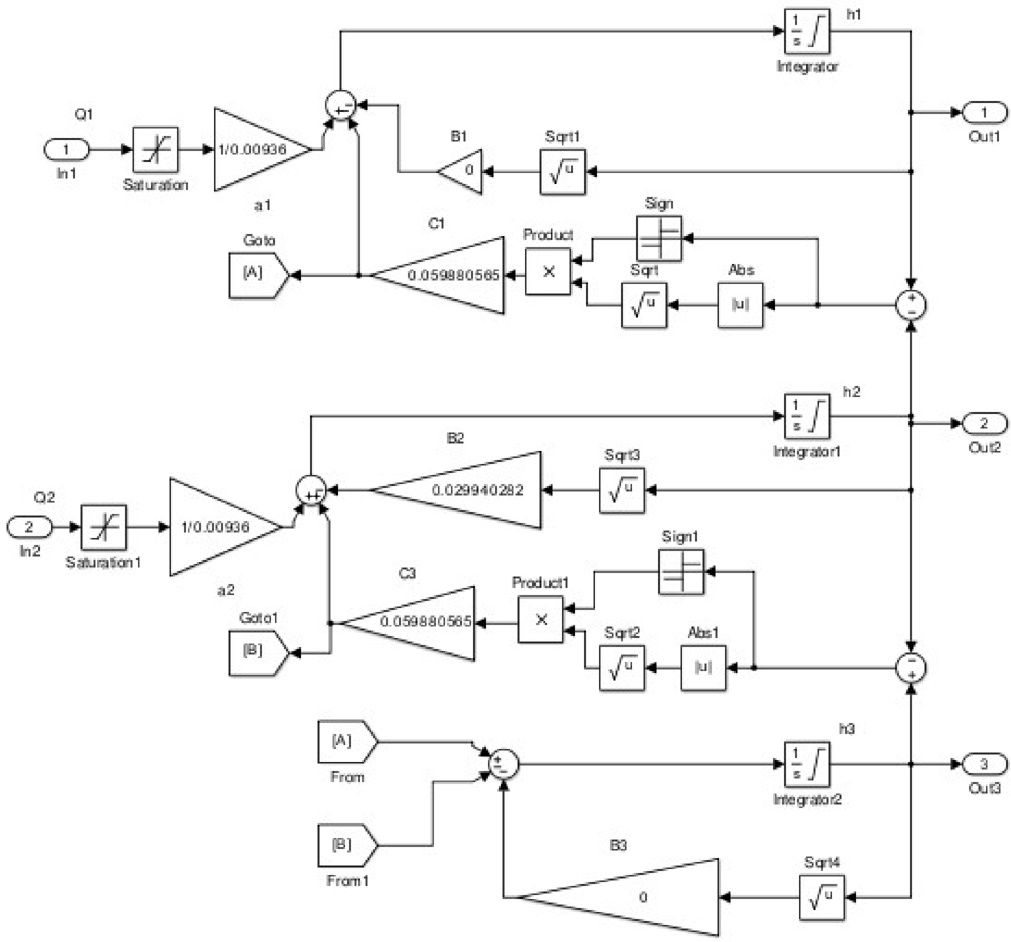

Figure 3.4: Plant model in Simulink

The plant model shown in figure 3.4 depicts the equations (4), (5) and (6). The value of B_1 and B_3 is zero as both the valves b_{z1} and b_{z3} are fully closed as shown in the figure 3.2. Rest of the valves are fully closed.

3.3.4.2 Open loop response for SISO

Below is the setup done on MATLAB Simulink to get open loop response of the system.

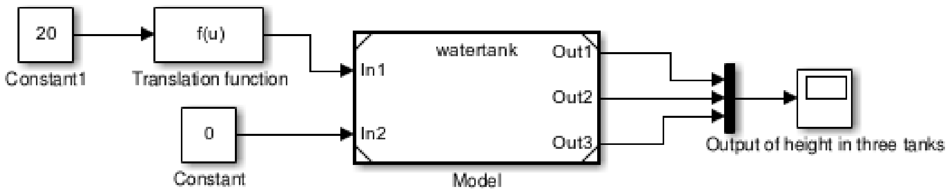

Figure 3.5: Simulink model for open loop response of the plant

Figure 3.6 shows the response of the system. Since water exits from tank 2 it has the lowest height and it enters from tank 1 hence it has the highest level of water of the three tanks.

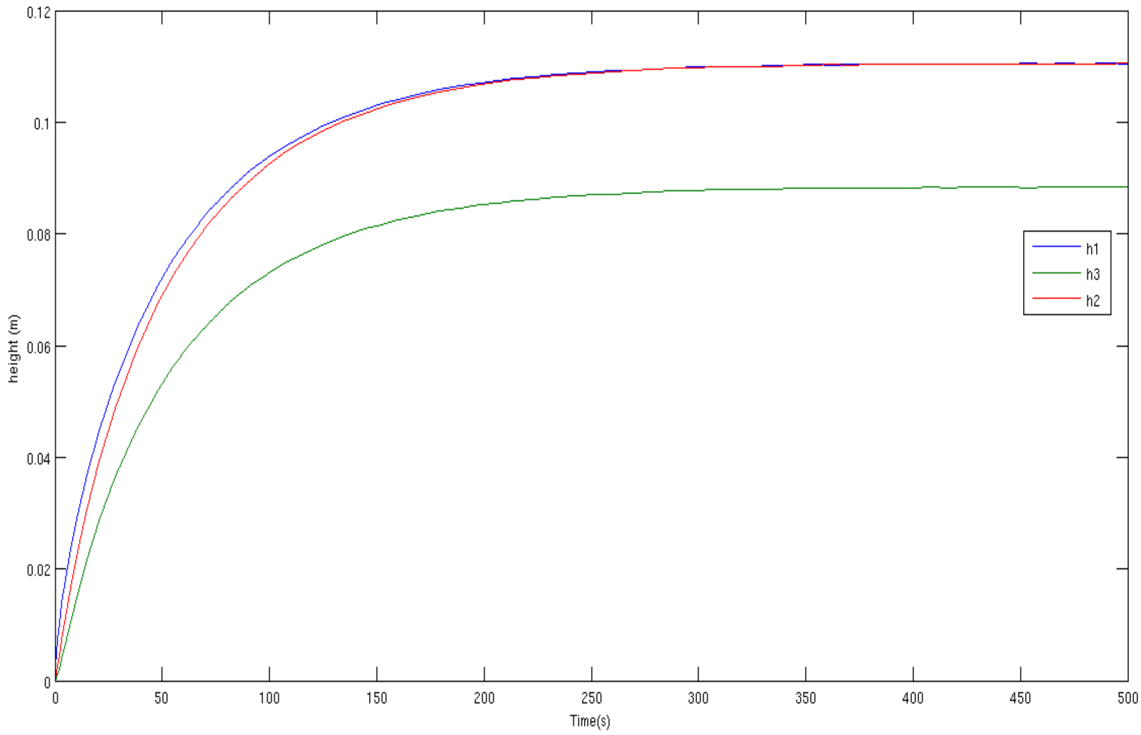

Figure 3.6: open loop response of plant in SISO configuration

3.3.4.3 Open loop response in MIMO configuration

Simulink model used for MIMO configuration is shown in figure 3.7.

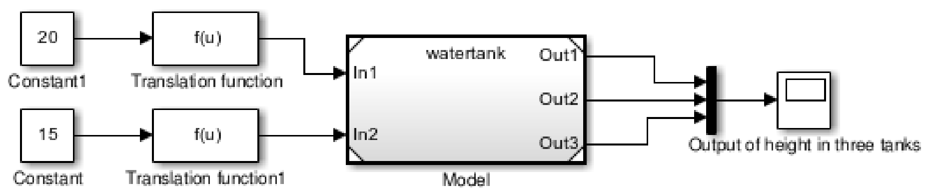

Figure 3.7: Simulink model for MIMO system

Response of the plant is shown in figure

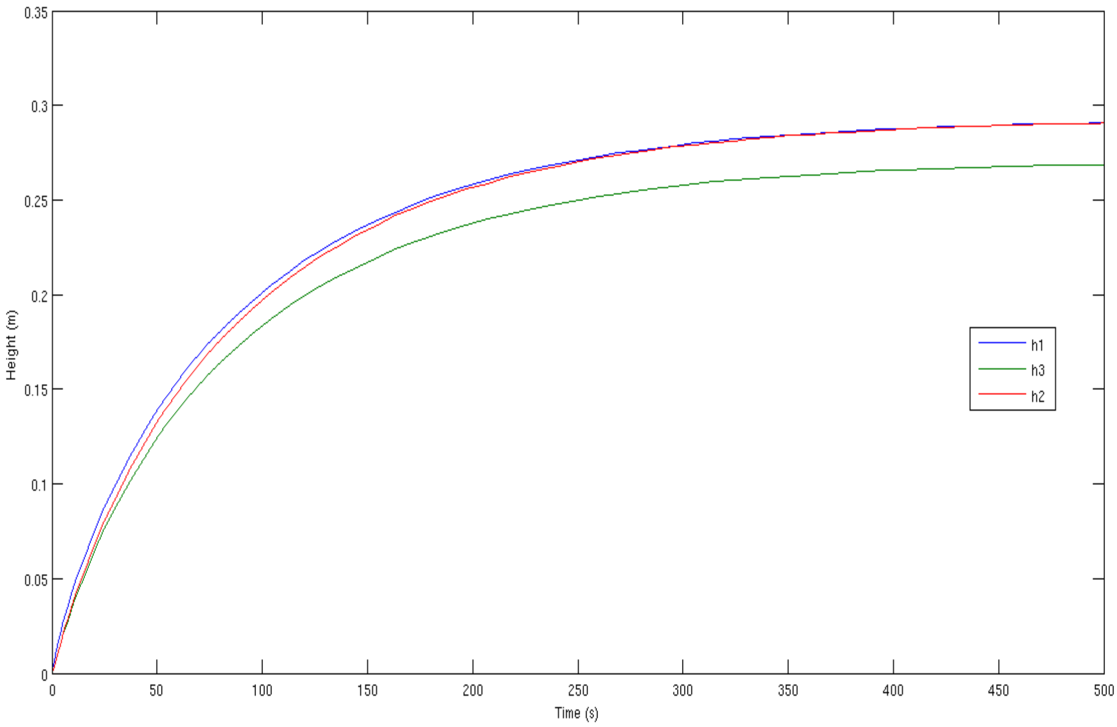

Figure 3.8: Open loop response for MIMO configuration

4 DESIGN AND DEVELOPMENT OF COUPLED TANK SYSTEM TESTING RIG:

The plant is a coupled tank where three tanks are coupled together. Each tank has its own outlet valve which drains the liquid from the tank to the reservoir. Further more the left most and the right most tanks have another valve connecting to the middle tank. These valves provide the coupling between the tanks.
A picture of the coupled tank system can be seen in figure 4.1

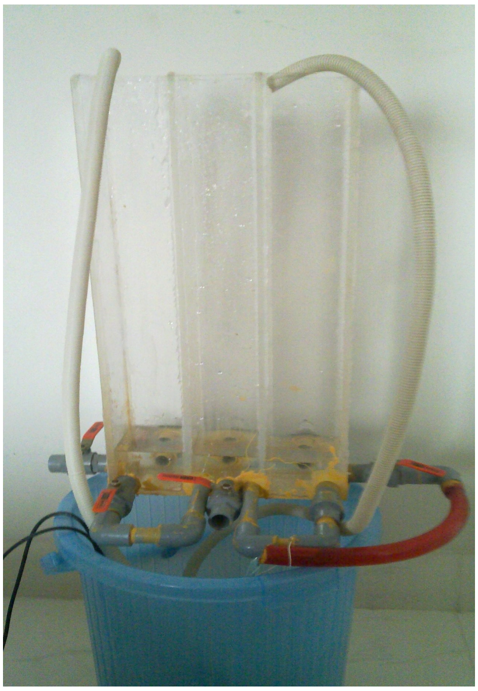

Figure 4.1: Coupled tank system.

4.1. Specification of tank:

Table 4.1 shows the specification of coupled tank system.

Dimensions	
Height	60cm
Width	30cm
Length	12cm
Dimension of each tank	60cm x 9cm x 10.4cm
Communication	
Pump speed control	UART
Level sensors	UART
Pump Speed control	
Range	0 – 100 (expressed as percentage of maximum flow rate)
Minimum step size for pump speed control	1 %
Level sensors	
Range	0 – 60m
Sensitivity	1 mm
Miscellaneous	
Number of tanks	3
Number of pumps	2
Number of level sensors	3

Table 4.1: Specification of tank

4.2. Tank Fabrication:

The tank was created by using acrylic pieces joined together by RTV silicone. This gave us three tanks separated by acrylic in between. Each tank has a height of 60cm. Holes were drilled in each of the three tanks so that PVC pipes with PVC ball valve can be used to couple the tanks together. Similarly drain valves were also connected as well as pressure sensor as shown in figure 4.2 and figure 4.3.

Figure 4.2: Pressure level transducers attached to the tank.

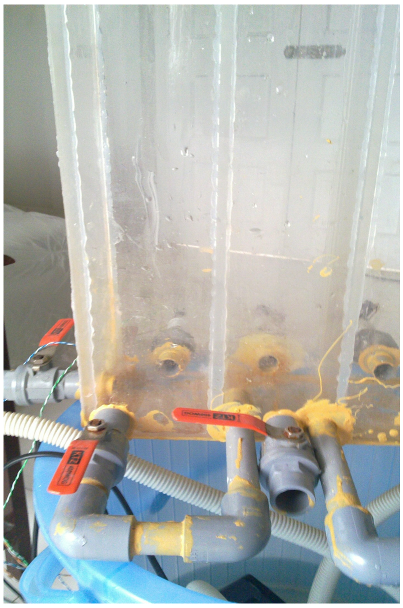

Figure 4.3: Ball valve coupling two adjacent tanks and drainage ball valve.

4.3. Level sensors integration:

Sensing level posed some problems and a multitude of sensing techniques were used until we found the one that gave the best results. Four approaches that were available are as follows:

1. Capacitive sensors
2. Ultra-sonic sensors.
3. IR distance sensors
4. Pressure sensors

4.3.1 Capacitive sensors

Capacitive sensors work on the principle of changing capacitance with the presence of dielectric. They have parallel plates or there can be 2 tubes, one inside the other. These parallel plates or tubes are dipped in the liquid whose height is to be measured. This then becomes two capacitor connected in parallel, one with the dielectric as air and the other with the dielectric as liquid and their ratio gives the level of the liquid.

Problems faced using this sensor.

These sensors are very sensitive and if the supply is not properly grounded even touching the probes introduces noise. Also the wires connecting the sensing capacitor with the measuring circuit has to be as short as possible else noise gets introduce in the readings and the level of the liquid would then be susceptible to change with the environment.

4.3.2 Ultra-sonic sensors

These sensors work on the principle of send sound waves and then receiving echo and measuring time it takes to complete the journey. Two peizo-electric transducers, one for sending sound waves and one for receiving sound waves, are placed on top of the tank to measure the level of water. To measure the distance a pulse is sent to the pulse input pin. This pin takes the trigger and sends 8 cycle of sound wave at 44kHz. After the

Figure 4.4: Ultra-sonic sensor

Problems faced using this sensor

These sensors have a very large angle of propagation which causes the sound waves to bounce of the walls of tank instead of the liquid. Secondly, since these sensors work on the speed of sound, they are not accurate if the environment is constantly changing. Humidity and temperature are two things which are not constant in our experiment setup, hence they were not suited here. The later cause can be compensated but the angle of propagation did not allow these sensors to be used.

Figure 4.5: Sharp IR (GP2Y0A02YK0F)

4.3.3 IR distance sensors

Sharp IR (GP2Y0A02YK0F) sensors were tested to measure the distance of the ball, which was floating on the surface of the liquid and was confined in a tube one end of the tube was in the water and the other end had the sensor mounted on it.

The sharp sensors use triangulation to detect the distance of the object. A pulse of IR light is emitted and the reflection of light is detected by the detector. These sensors have a small linear CCD array to detect the angle of the reflected light. Since the angle of the reflected light varies with the change distance, it quite accurately calculates the distance with the measured angle.

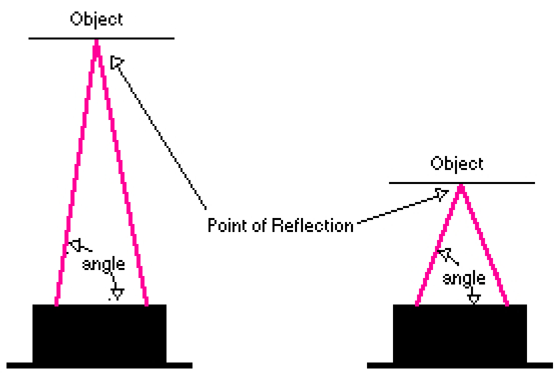

Figure 4.6 Sharp IR sensor detection technique.

This method is almost immune to interference from ambient light and also the distance reading is not affected by the color of the object being detected.

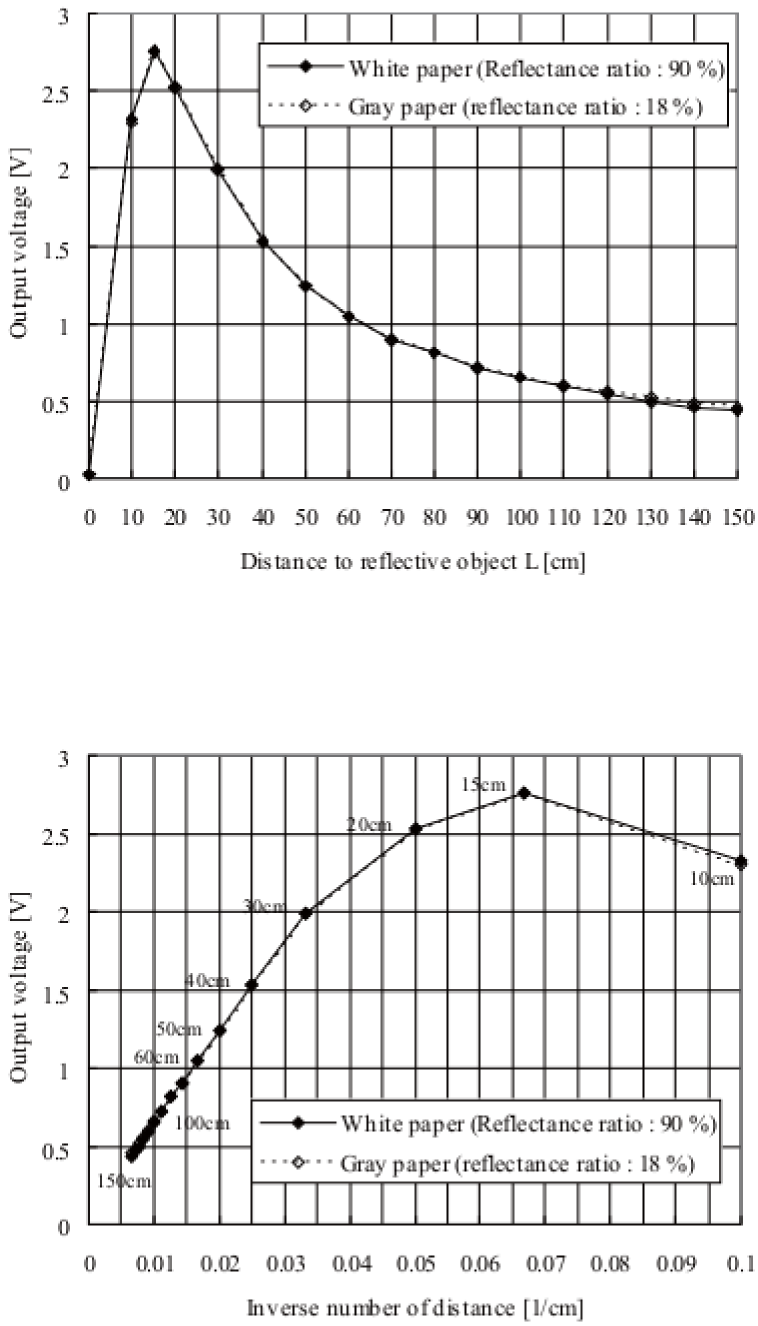

Figure 4.7 Sharp IR sensor distance relationship with voltage

Problems faced using these sensors.

First their output was not linear, instead there is a curve on the data sheet which showed the relationship between voltage and distance.

Since the inverse of distance has a straight line till 30cm it was used to convert voltage to distance but this gave us a dead zone of 30cm which was taken care of by using a longer pipe (90cm).

Second problem we faced using this sensor was that it has a step size of 7mm. The output of the sensor became constant and only jumped at 7mm interval. Also when it was near to 7 mm jump interval it would start oscillating. Table 4.2 shows the real and measured values to get a better understanding:

Actual distance(cm)	Distance measured by Sharp IR sensor(cm)
50.1	50.1
50.2	50.1
50.2	50.1
50.2	50.1
50.3	50.2
50.3	50.1
50.3	50.1
50.4	50.1
50.4	50.1
50.5	50.1
50.5	50.0
50.5	50.1
50.6	50.1
50.6	50.2
50.6	50.1
50.7	50.1
50.7	50.1
50.7	50.8
50.8	50.1
50.8	50.8
50.8	50.8

Table 4.2: Experiment values of sharp IR sensor

As it can be seen that when the value was incrementing from 50.1cm to 50.8cm there were some oscillations which are not needed and cause problems.

4.3.4 Pressure Transducers

Figure 4.8:Pressure transducers

Pressure transducers were also tested for sensing the level of water in the tank. Pressure transducers we had have 4-20mA output and could measure up to 6bar of differential pressure. In our tank application we only needed to measure up to 0.1bar. In order to maintain a 1mm accuracy we either needed a 16bit Analogue to Digital converter (ADC) or we had to amplify the incoming signal so as not to loose precision. Amplifying the signal also amplified the accompanying noise which made the matters even worse hence it was decided to use 16 bit ADC. Since the controller we had could only convert analogue signal with 10 bit resolution we used a signal processing techniques which can increase the resolution of ADC. This signal processing techniques is known as 'Oversampling and Decimation'.

A method for increasing the effective resolution of the ADC is oversampling and decimation. This technique involves oversampling of the input signal so that a number of samples can be used to compute a virtual result with greater accuracy than a single real sample can provide.

For low-bandwidth signals such as temperature measurements, static pressure and power supply voltages, accuracy of measurements can be increased using oversampling and decimation. Several criteria must be met in order for this technique to give desired results[14][15].

1. The signal of interest must not change more than 0.5 LSB of the final result during sampling interval. For example if we have a 12bit ADC and we want a resolution of 14 bit then the signal of interest must not vary more than 1/8 LSB of ADC.
2. During the sample interval, the ADC must perform the conversion of the

signal 4^n times, where n is the number of virtual bits desired in the result. Here n is taken 6.

3. Noise must be present in the input and it must have an amplitude greater than 1 LSB of the ADC; it also must have an mean value of zero, and it must be randomly distributed (white).
4. The result is truncated by shifting right 2^n places to yield the desired resolution.

Since we are using pressure transducers and we are measuring at 16 bit resolution, the change in actual reading from pressure transducer has to be less than 1/32 LSB of ADC.

Since we have a 10bit ADC with reference voltage equals 2.56V, we have:

$$1/128\, LSB\, of\;\; ADC=\frac{2.56}{1024*128}V=19.53\mu V \tag{27}$$

With unity gain and a shunt resistor of 200Ohms we have a difference in current output of transducer equals to:

Difference in current output of transducer

$$\Delta i=\frac{19.53}{200}=97.65\, nA \tag{28}$$

we have a pressure transducer which can measure up to 6bar of pressure and has an output in the format of current loop, 4-20mA, this corresponds to a total difference in height which is acceptable in this scenario is:

$$\Delta h=\left(\frac{97.65\times 10^{-6}\cdot 6}{16\times 0.09807}\right)\approx 0.4\, cm \tag{29}$$

since 0.09807 bars = 1m of water depth.

This change is acceptable in one reading of ADC. Since we can have 3 readings per second this corresponds to 1.2cm of change in a second, which is well within our limits. The actual change measured per second is 1.0cm maximum.

Now for 16bit resolution we had to take 4^6 samples, which are equal to 4096 samples. At 200kHz we get approximately 3Hz of bandwidth.

For the white noise there was already enough noise in the circuit which was measured up to 5 times the LSB of ADC this adequately full filled our purpose.

To counter the DC offset which presents itself due to oversampling the sensors have to be calibrated before use which can be done easily when the unit is turned on and there is no water in the tank whatever the reading is, is the actual offset.

At the end the result is truncated by shifting the value right 6 times.

Interface circuit description:

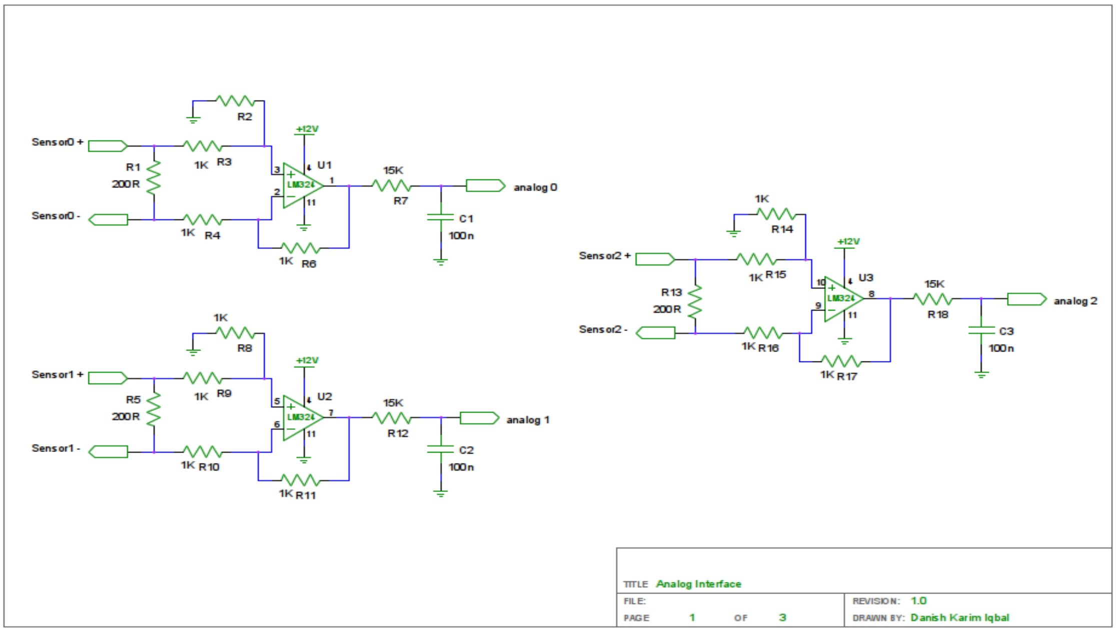

Figure 4.9: Pressure transducers interface circuit

Figure 4.9 shows the schematic for interfacing pressure transducer to the micro-controller. Here R1, R2, R3, R4, R5, R6, R7, U1 and C1 together make the interface for sensor 0, which is located at left most tank. Similarly R8, R9, R10, R11, R12, R13, R14, U2 and C2 are for sensor 1, located at center tank, and R15, R16, R17, R18, R19, R20, R21, U3 and C3 are for sensor 2 which is mounted on right most tank. The op-amps are connected in differential input arrangement and have a gain of one. The 200R resistor is the shunt resistor this gives us a voltage range of 0.8V to 4V.

Since the actual voltage range we are measuring varies from 0.8V to 0.9V it is enough for us.

After the measurement there is a low pass filter with cutoff frequency at approximately 106Hz. This was so that it would allow the 50Hz AC noise to enter the circuit and filter out all spikes and high frequency noise. This noise was necessary for oversampling and decimation.

4.4. Pump Integration:

Small pumps are hard to find especially DC pumps. After searching we were able to find a DC and some AC pumps used in aquariums. Both the DC pump and the AC pumps used the same technology to pump the liquid, both were centrifugal pumps. The underlying technology for the prime mover is same in both the pumps. It was assumed that DC pumps might have DC motors but upon inspection we found it otherwise. DC motors had the same motor as AC, permanent magnet synchronous motors, and used switching circuits to produce AC voltage for driving the motor from DC current. This especially posed a problem because now controlling the speed of impeller was difficult as there was no access to the circuit because it was potted to have an IP68 rating. With AC pumps, varying the frequency of driving voltage varied the speed of impeller, hence they were used in this model.

Pumps used in this plant are submersible aquarium pumps. These pumps have permanent magnet rotor which rotates due to the rotating magnetic field generated by AC currents in the stator. The speed of the rotor is directly proportional to the frequency of ac voltage present at stator. So to change the flow rate of the pump its frequency was varied. Since the pumps worked at 220VAC in order to change its frequency an inverter was made whose frequency was made variable using a micro-controller.

Interface circuit description

Figure 4.10 depicts the schematics of the circuit for driving the pumps.

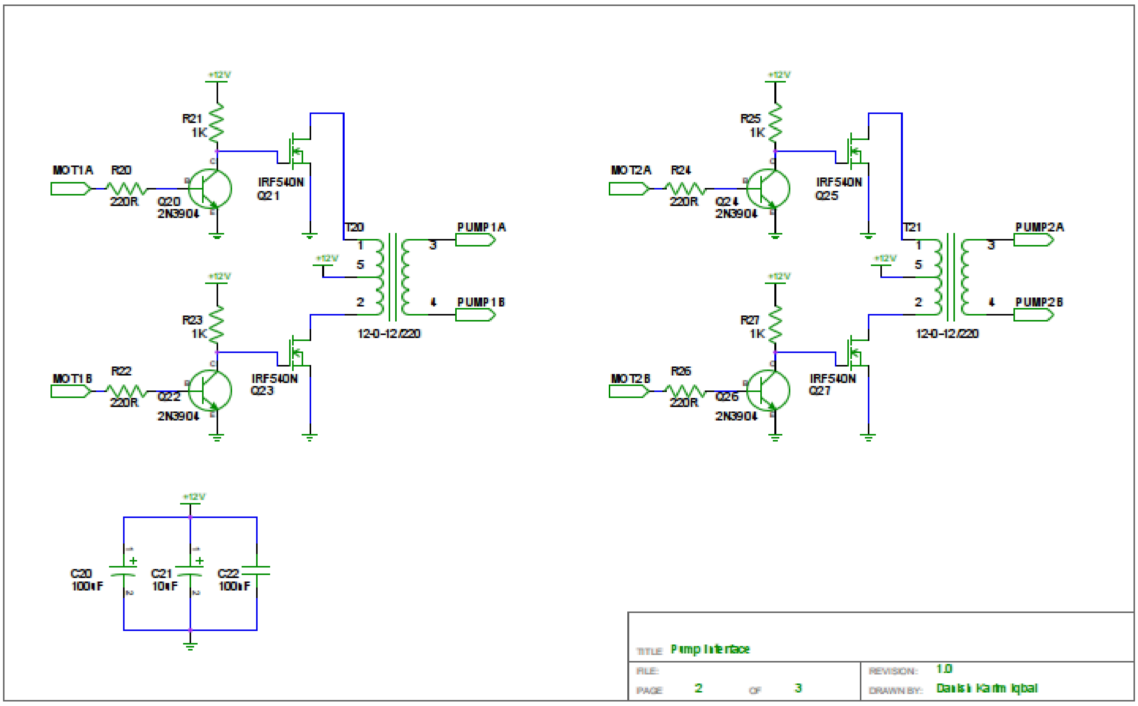

Figure 4.10: Pump interface schematics

In the circuit shown in figure 4.10, Q20 and Q22 are buffers for driving MOSFETs, Q21 and Q23. The input signal comes from micro-controller and is of the magnitude 0V, for low logic level, and 3.3V, for high logic level. These have to be amplified to 12V and 0V. Q20 and Q22 are inverting buffers, the logic level was inverted to get the desired MOSFET to turn on or off. Transformer, T1, is a step up transformer which has primary side rating of 12-0-12 volts and secondary side rating of 220 volts. This transformer provides the driving voltage to drive the pumps. Q20,Q21,Q22, Q23 and T1 are responsible for driving pump 1 similarly Q24,Q25,Q26,Q27 and T2 are responsible for driving pump 2.

Signals from micro-controller are such that the output frequency of the pump 1 varies from 29.67 Hz to 51.81 Hz and output frequency of pump 2 varies from 47.39 Hz to 61.35 Hz. The difference in frequency is due to the fact that 2 pumps of different model and make are used. It was by experiment that their optimal frequency range was found and programmed.

In order for the pumps to operate at different speeds they had to be modifies. The modification was done to the impeller. Impeller by default had a slip so they can lock their speed at 50Hz without the need of gradually increasing the speed but this posed a problem as with lower speeds this slip caused the impeller to have a hammer like effect where as soon as it started rotating it came in contact with full load of impeller and it caused the rotor to stop momentarily and start all over again. To correct this impeller the slip in impeller was finished but now, to run at full speed, speed had to be gradually increased or decreased. This was done by setting the set point in micro-controller and the micro-controller would increment the frequency 0.22Hz at every 90ms for pump 1 and 0.14Hz at every 90ms for pump 2.

4.5. Micro-controller board

The micro-controller board used is ArduinoMega2560 its specification is in table 4.3.

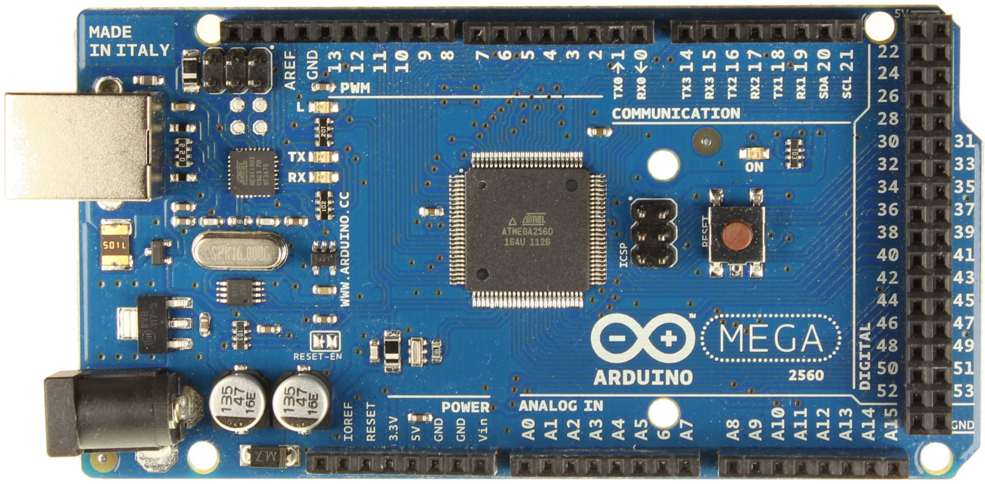

Figure 4.11: ArduinoMega2560 front view

Micro-controller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by boot-loader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Table 4.3: Specifications of ArduinoMega2560

The connection for micro-controller board is shown in Figure 4.12. For sake of simplicity only the pins used in Arduino are shown.

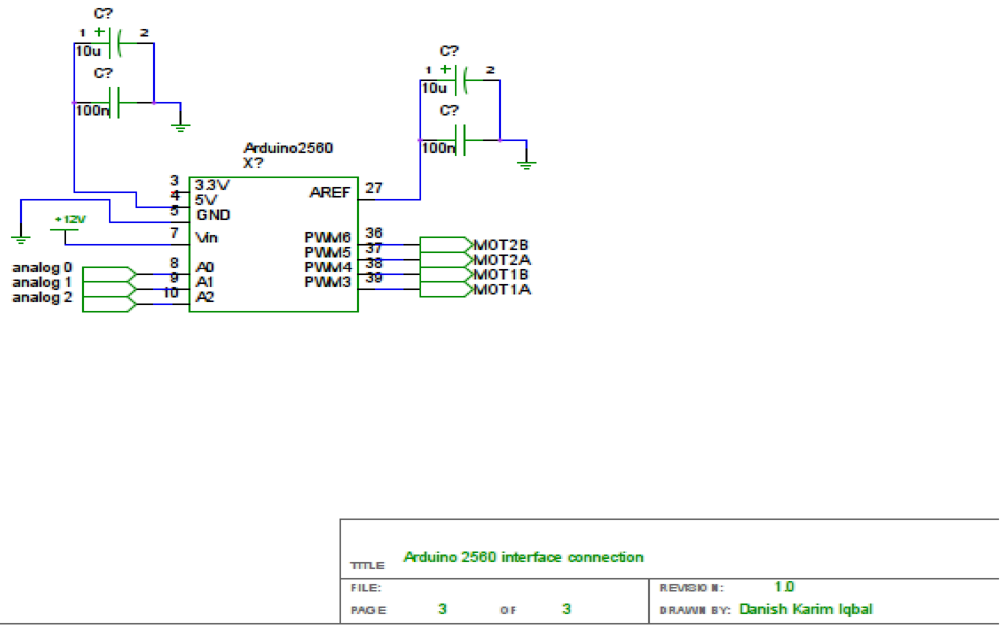

Figure 4.12: Schematics of Arduino interface with other modules

4 digital I/O pins are used as output and are responsible for driving the MOSFETs and three analog pins are for interfacing with the sensor interface circuit.

A bypass capacitor is also connected between AREF pin and ground to bypass all high frequency noise to ground. This is necessary as readings the Voltage reference was not stable and the ADC conversion was also not accurate.

Vin is 12 Volts to power up the board and other peripherals such as MOSFETs and pressure transducers.

Frequency generation

Frequency was generated by the help of an 8 bit timer available on the micro-controller. This timer has a configurable prescaler. This prescaler divides the main clock by 1,2,8,64 and 128. e.g., If we have a clock of 16MHz and we use a prescaler of 8, then we would have a clock of 2MHz for the timer. Timer was set to interrupt at every 50us and two state machines were made each for their respective motor. Each state machine would return to its initial state after completing a cycle. The cycle timings were calculated and are stored in the flash memory.

Analog readings

Apart from the interrupt there are two functions running continuously. One is for analog reading and one is for communications. The analog function reads the analog value of each of the three sensors and stores them in variable. Once 4096 values are accumulated the value is then transferred to another variable which will send the data after every second.

Communication

Communication to and from the PC is done using Universal Asynchronous Receiver and Transmitter(UART). This UART is configured at 57600 baudrate, 8 bits, no parity bit and one start and one stop bit. This UART is then connected to Atmega8u, found on ArduinoMega2560 board, which performs necessary USB to UART conversion. Communications function have two jobs one is to listen for input commands and the other is to transmit data every second. A flag is set after every second on which the function transmits the data and clears the flag.

On every iteration this function also checks the input buffer. If there is data in input buffer and the receiver is timed out then it checks if there is a valid command in the buffer. Valid command corresponds to the command which changes the frequency of the pumps. Once the valid command is received the values of each pump is updated accordingly.

Protocol for communication was kept simple. At every second the microcontroller would send data at 57600 baud in the following format

<adc value 1>,<adc value 2>,<adc value 3><enter>

e.g., following are two data sets sent by the micro-controller to the PC.

124554,123454,124485

124554,123454,124485

To receive commands from PC a simple format was used to send the values of pump to the micro-controller. The format is as follows:

MOT(<value for pump 1>,<value for pump 2>)<enter>

e.g., if we wanted to set the value of pump 1 to 50% and pump 2 to 30% we would simply send the following

MOT(50,30)

the terminating character is a new line or 0x0D ASCII code.

4.6. Interface with MATLAB

MATLAB has many tools and libraries to connect external hardware. This hardware can be from any company that makes lab systems in collaboration with MATLAB or individuals who want to use the power of MATLAB on their hardware.

MATLAB instrument control toolbox provide the necessary interface to connect many types of hardware. A brief list of interface is as follows:

1) Serial interface
2) TCP/IP interface
3) UDP interface
4) GPIB
5) VISA USB

Since we had a serial port readily available hence it was our first option. In MATLAB the plant was configure as follows

1. We add two blocks from Simulink library, Instrument control toolbox, “query instrument” and “to instrument”.
2. Configure “query instrument” so that it can receive data from micro-controller board.
3. Configure “to instrument” so that it can transmit data from PC to micro-controller.

Figure 4.13: "Query Instrument" and "To Instrument" blocks found in Instrument Control Toolbox

Configuring Query Instrument block

1. Figure 4.15 shows Instrument Initialization screen. If the hardware requires an initialization string it can be entered here. Since our hardware does not require one we have set the value to none.
2. Hardware Configuration tab allows to configure the hardware. Interface can be chosen here and it can be configured here. We are using serial with 57600 baud rate as shown in figure 4.14.

3. Query tab is for querying the instrument for data as shown in figure 4.16. In our case the micro-controller automatically sends the data hence the query string is null. ASCII format string is to tell in what format the data is for our case its as follows:
   <integer>,<integer>,<integer>
   this is why the entry is '%d,%d,%d'. It has to be frame output or else it will only transmit one value at a time.

To instrument block:

1. Figure 4.17 shows Instrument Initialization screen. If the hardware requires an initialization string it can be entered here. Since our hardware does not require one we have set the value to none.
2. Hardware Configuration tab allows to configure the hardware. Interface can be chosen here and it can be configured here. We are using serial with 57600 baud rate as shown in figure 4.17.
3. Send tab is for specifying the format in which the data from Simulink is sent to the instrument. It supports binary and ASCII format and we have chosen ASCII format. Then a format specifier string is entered in the way the data is sent. In our case the data that is to be sent should be in the following format:
   MOT(<value for pump1>,<value for pump 2>)
   after the data an new line command must be sent of 0x0D is to be sent. We have used '\n' for new line which is an escape character provided by MATLAB for new line.

Once the settings are completed a controller can be connected to 'Query Instrument' and 'to instrument' block and simulation can be started.

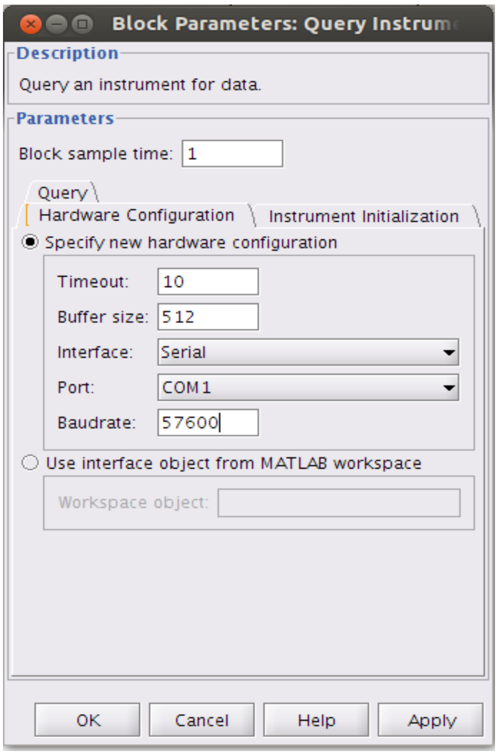

Figure 4.14: Hardware configuration in query instrument block

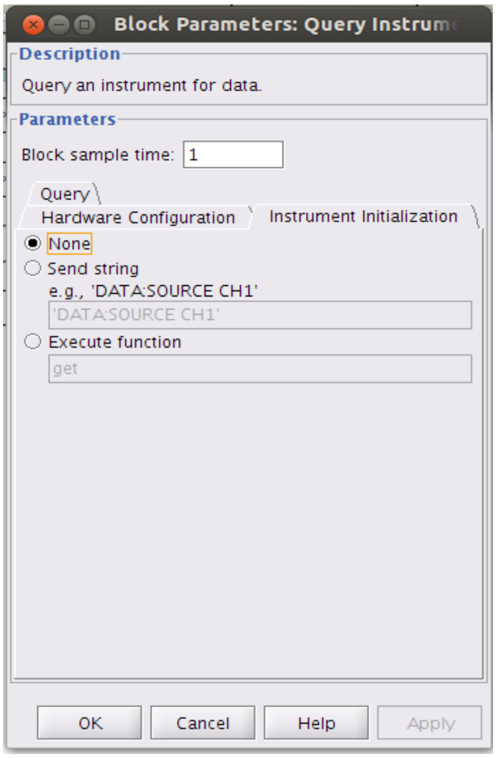

Figure 4.15: Instrument initialization in query instrument block

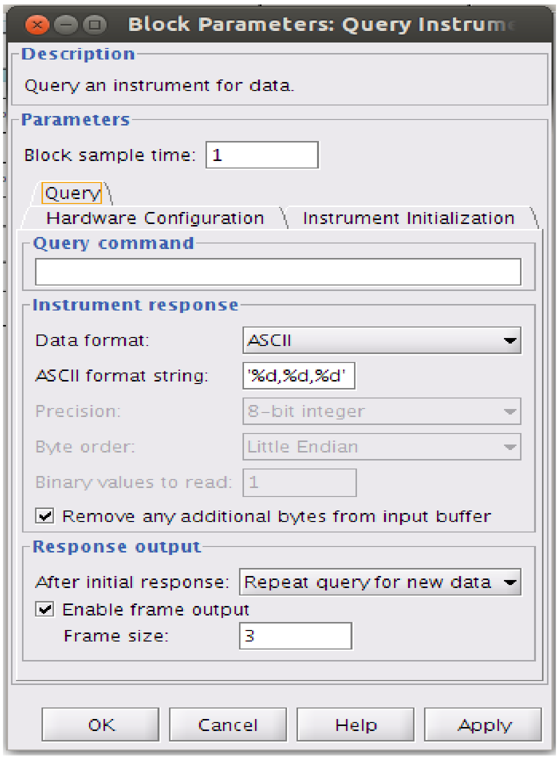

Figure 4.16: Query tab in query instrument block

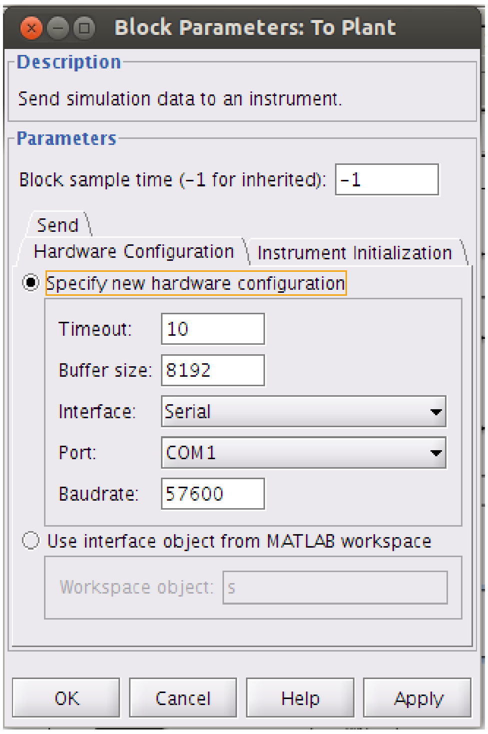

Figure 4.17: Hardware configuration in to Instrument block

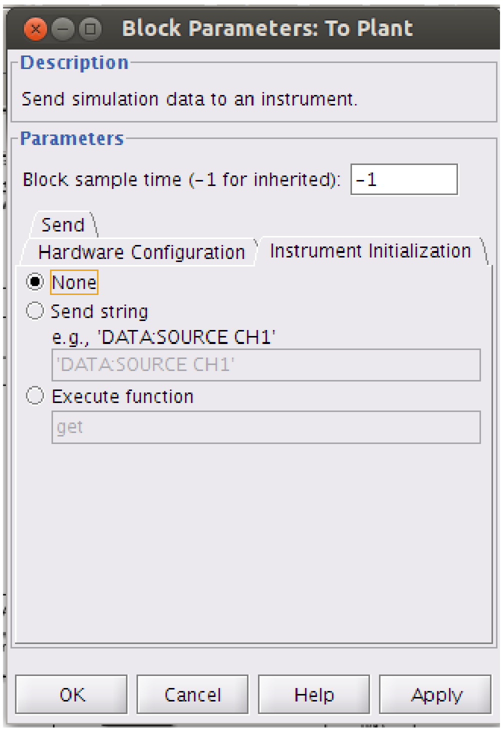

Figure 4.18: Instrument initialization tab in to instrument block

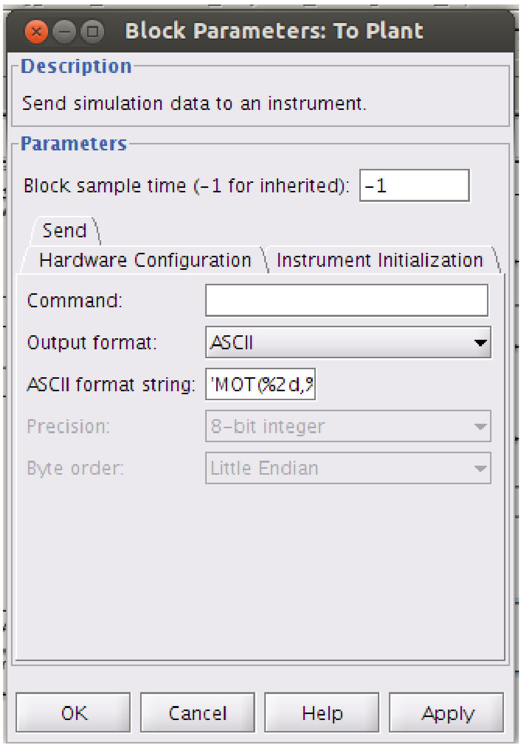

Figure 4.19: Send tab in 'to instrument' block. It configures the format in which the data is sent.

4.7. Open loop response in SISO configuration

For open loop response the tank was setup as shown in figure 3.3. A Simulink model with no feedback was used and the output of the plant was recorded which is shown in figure 4.20.

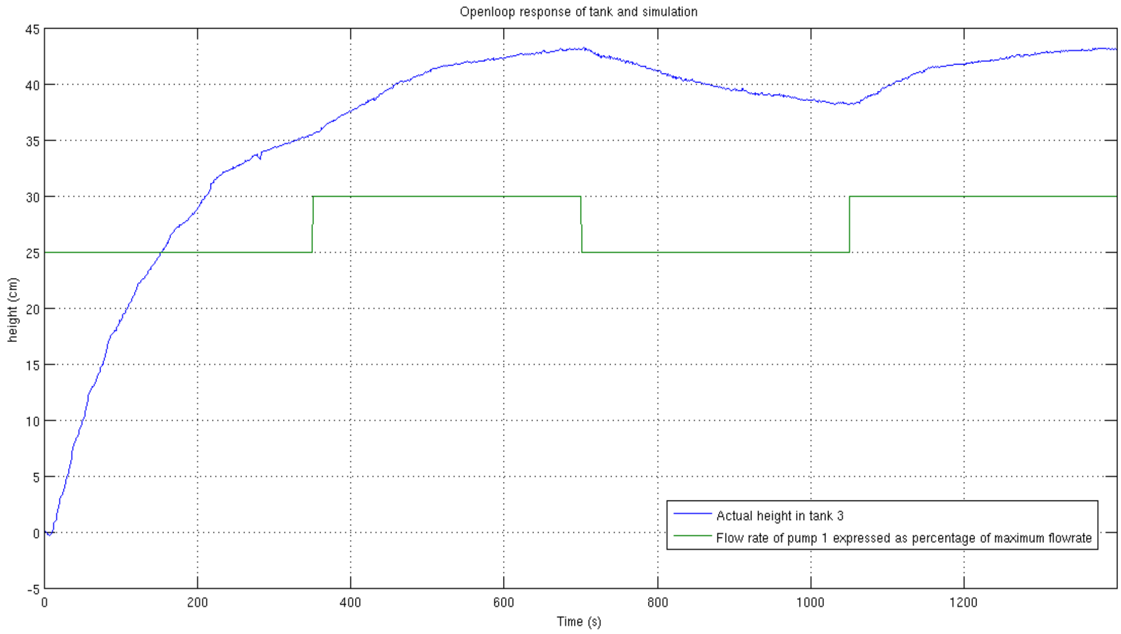

Figure 4.20: Open loop response of tank in SISO configuration. The level of liquid recorded is in tank 3 while pump1 was set as a percentage of its maximum flow rate.

The response shown in figure 4.20 is the level change in tank 3 due to water flowing in from tank 1 by pump 1. All the valves are fully closed except the leakage valve in tank 2 which is 50% open.

4.8. Open loop response in MIMO configuration

For open loop response of MIMO system the tank was setup as shown in figure 3.2. The tank was connected with no feedback. Only the percentage flow of pumps was set and the level in tank 2 and tank 3 recorded. As shown in figure 4.21.

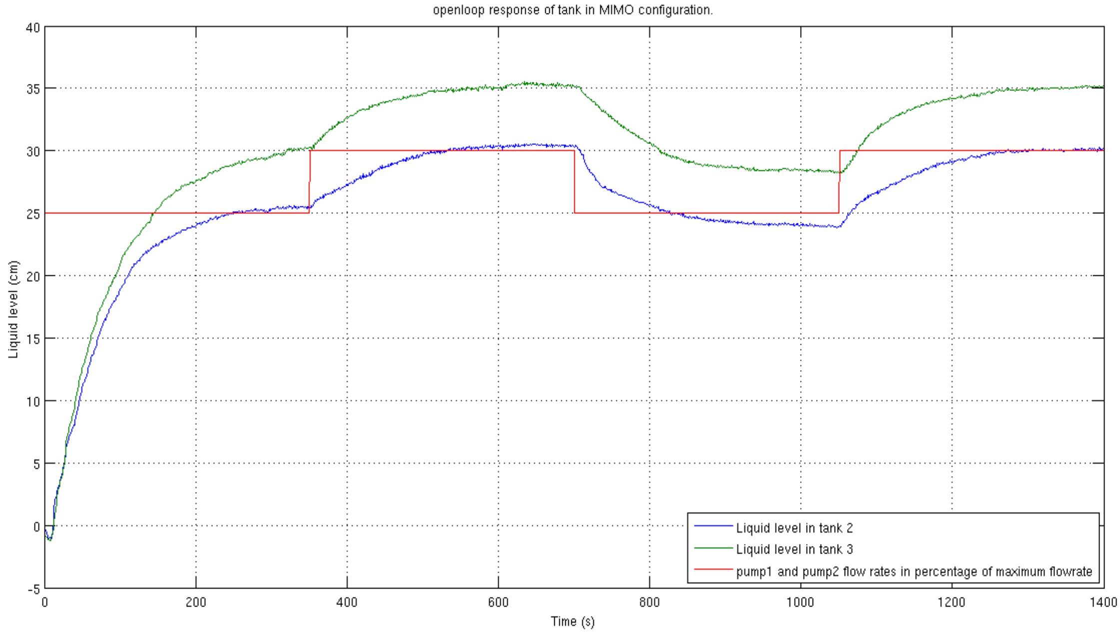

Figure 4.21: Open loop response of tank in MIMO configuration. Levels in tank 2 and tank 3 are recorded while pump1 and pump2 have a flow rate which is a percentage or their maximum flow rate.

4.9. Comparison of open loop response of simulation and actual plant.

Figure 4.22 shows the comparison of actual tank with the mathematical model derived in Chapter 3. It can be clearly seen that there are errors in the mathematical model.

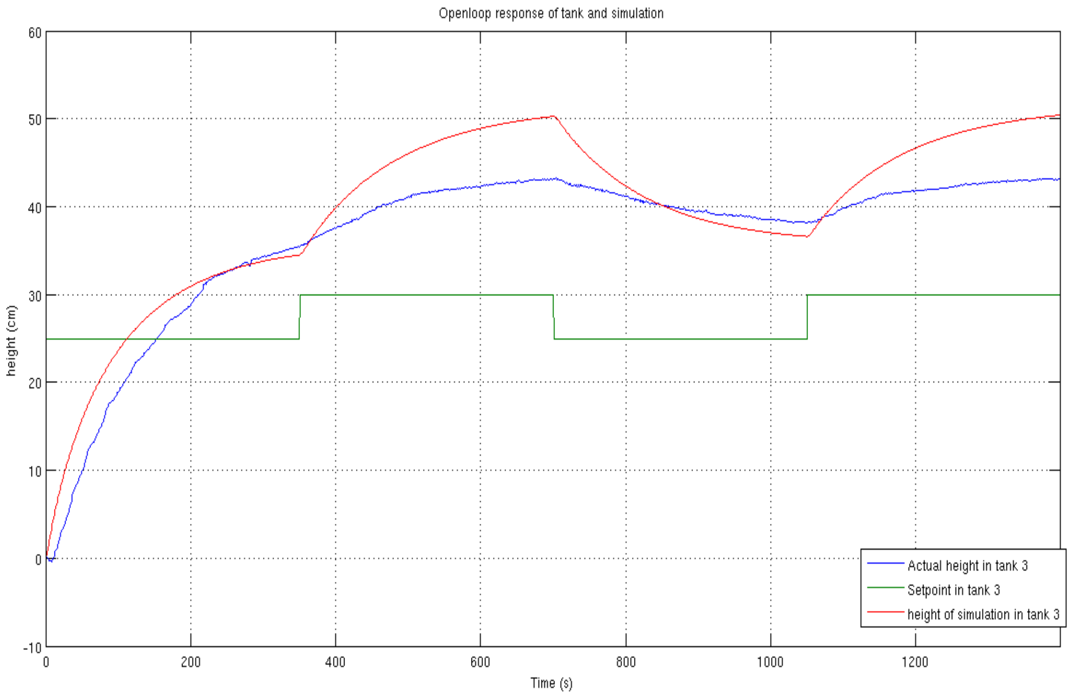

Figure 4.22: A comparison of mathematical model with an actual plant

At the start of the curve, height in actual plant has a delay, dead time. This is not handled in the mathematical model. Further more the pumps have variable flow rate depending upon the level of water in the reservoir tank. To overcome this a large reservoir can be used or the pumps can be mounted on the float which would ensure constant depth of pump in water.

5 Control performance of coupled tank system using different control schemes.

5.1. Introduction

This chapter discusses the coupled tank system's behavior when different control schemes are applied to it. In the first part we will attempt to stabilize height in tank 3 using PID. For PID the gains would be calculated using Ziegler-Nichols rules. In the second part we will use an adaptive fuzzy controller to maintain the level in tank 3. The set height in both the cases would change to see the control systems reaction.

5.2. PID control scheme

5.2.1 Open loop response of the system

To get the open loop response of the system we setup the tank with parameter as shown in table 5.1. A Simulink model was created where a step input was given to the plant and the response was noted. Once the response was stable another step input was applied and the level change in tank 3 was recorded. Step input parameters are shown in table 5.2. The tank configuration is same as in chapter 3, figure 3.3

Parameter	**Value**
a_{z1}	Fully open
a_{z2}	Fully open
b_{z1}	Fully closed
b_{z2}	Fully closed
b_{z3}	Open 50%
Control parameter	h_3
Output	Pump 1

Table 5.1: Setup values of tank for PID control scheme

Property	**Value**
Step 1 duration	300s
Step 1 value	15
Step 2 duration	300s
Step 2 value	25

Table 5.2: Step input characteristics to the plant

The response of the system can be seen in figure 5.1.

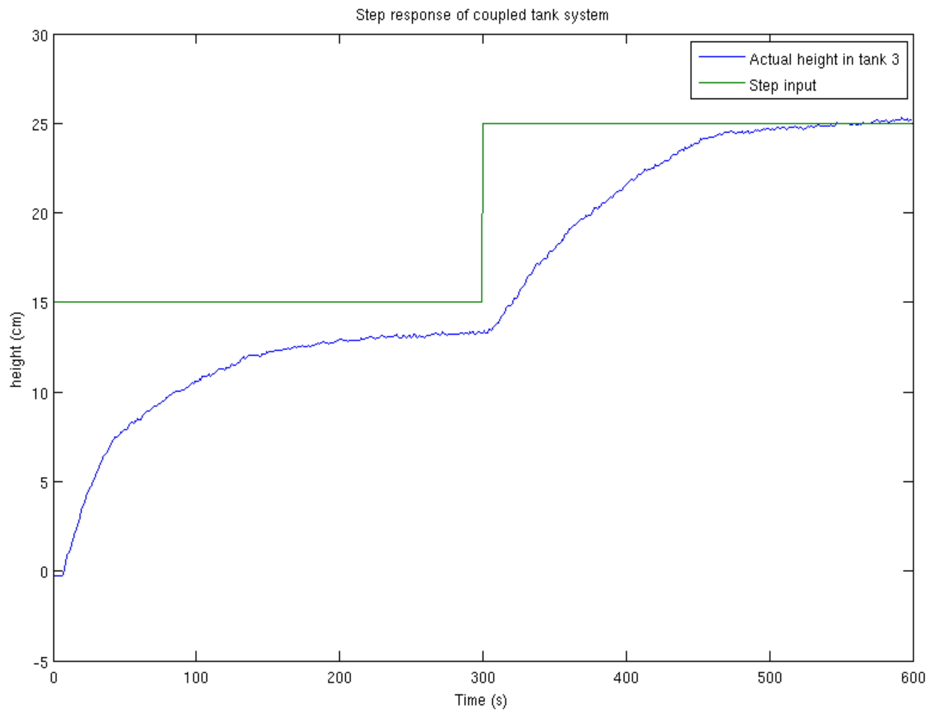

Figure 5.1: open loop response of coupled tank system

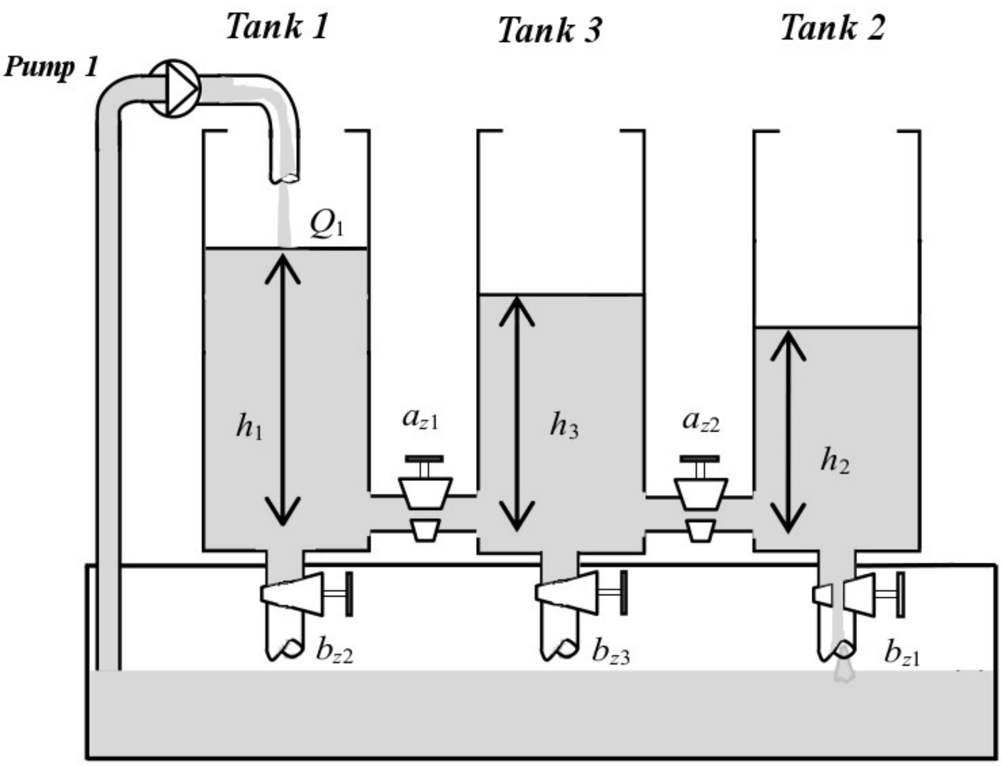

Figure 5.2: Tank configuration for PID and adaptive controller.

5.2.2 Ziegler-Nichols tuning

5.2.2.1 Introduction

In the 1940's, two empirical method were devised by Ziegler and Nichols to obtain controller parameters[16]. In scenarios where the system were non-first order and had large dead times their methods were used and involved intense manual calculations[17]. With the advent of powerful computers and improved optimization software, these manual methods are used scarcely but still the following two methods are still employed today[18]

- Ziegler-Nichols closed-loop tuning method
- Ziegler-Nichols open-loop tuning method or process reaction method

Both of these methods tune PID controller shown in

$$u(t)=K_c\left(\varepsilon(t)+\frac{1}{\tau_i}\int_0^t \varepsilon(t)\,dt\right) \tag{30}$$

Ziegler-Nichols closed-loop tuning method

Using the closed-loop method we first determine the critical gain and ultimate period. They are calculated by following the following series of steps:

1. Firstly note whether the required proportional gain, K_c, is negative or positive.
2. Turn off integral and derivative part of the controller. Make T_i infinity and T_d equals zero.
3. Now slowly turn up or down (if K_c is positive then turn up else turn K_c down) the K_c. This is done in step increments and after every increment wait till the steady state.
4. Keep changing K_c till you get sustained oscillations. This gain is the critical gain, K_u , and the period of oscillation, P_u , is referred to as ultimate period.
5. Using K_c and P_u the controller gains can be found using table 5.3.

Control Type	K_p	T_i	T_d
P	$0.5K_u$		
PI	$0.4K_u$	$0.8P_u$	
PID	$0.6K_u$	$0.5P_u$	$0.125P_u$

Table 5.3: Closed-Loop calculations of K_p , T_i and T_d.[18]

Advantages

1. Easy to experiment. Only proportional gain is chained.
2. Whole system dynamics is included and it gives a more accurate picture of system behavior

Disadvantages

1. Experiment can get time consuming.
2. Some times the system can go into unstable state.

Ziegler-Nichols open-loop tuning method or process reaction method:

Using the open loop response, such as shown in figure 5.1, we can calculate PID gains for that particular set-point this method is known as Process Reaction method. In this method, a system is already in place and its variables are measured. At first steady state is achieved in the system, and then a disturbance, X_o, is introduced. To get the curves as shown in figure 5.1 the system has to be open loop system and the disturbance is a step input change in set point. This curve, which is produced in response to a step input, helps us in determining several parameters which are shown in table 5.4.

Symbol	Meaning
M_u	Value of the system when it reaches steady state after disturbance
X_o	The magnitude of the step change
τ	Time for the response to occur
τ_{dead}	Dead time or transportation lag.

Table 5.4: Parameters from a process reaction curve and their meaning

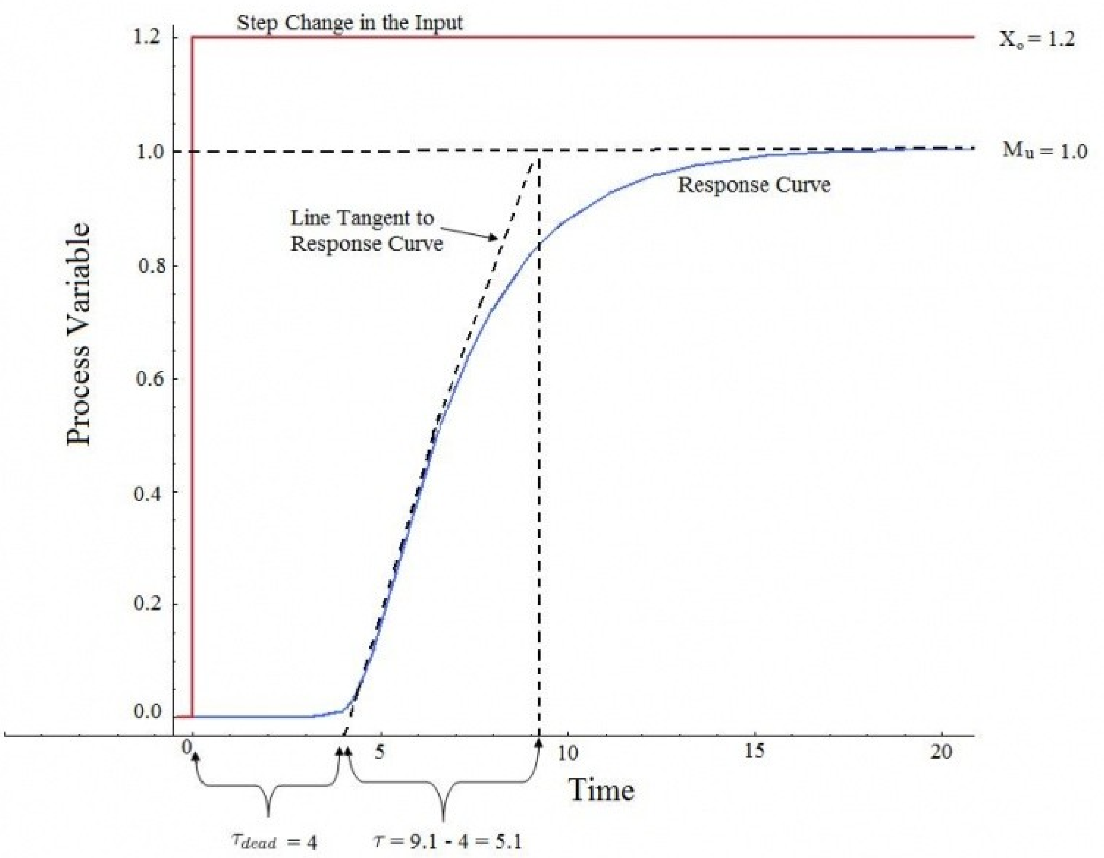

Figure 5.3: An example of calculating parameters M_o, X_u, τ and τ_{dead}.

In order to use Ziegler-Nichols the following steps must be performed:

1. Turn off the feedback of the system to have it in open loop state and apply a step input to it.
2. Once the system reaches steady state we can then determine the parameters shown in table 5.4. An example of a system's response is shown in figure 5.3. to find τ and τ_{dead}, a tangent to the response curve is drawn at the point of inflection and then the values can be found out as shown in figure 5.3.
3. Now the reaction rate is calculated using (31).

$$K_o = \frac{X_o \tau}{M_u \tau_{dead}} \qquad (31)$$

4. Once we have reaction rate and time lag we can plug the values in Ziegler-Nichols open-loop tuning equation for relevant controller (P,PI,PID) to get the gains.

	K_c	T_i	T_d
P	K_o		
PI	$0.9K_o$	$3\tau_{dead}$	
PID	$1.2K_o$	$2\tau_{dead}$	$0.5\tau_{dead}$

Table 5.5: Open-loop calculations of K_c, T_i and T_d [18]

Advantages

1. very easy and quicker than other methods
2. robust and very popular
3. out of the two techniques, this method is least disruptive and easiest to implement.

Disadvantages

1. It depends only on proportional gain to estimate integral and derivative gain.
2. Approximations for the gains may not be accurate for different systems
3. PD, I and D controllers cannot be tuned this way

5.2.2.2 PID tuning for 15cm set point

Figure 5.4 shows parameter extraction necessary to implement Ziegler-Nichols rules. A tangent to the curve was made and then four parameters in total were calculated as shown in table 5.6

Parameter	**Value**
M_u	13.24
X_o	15
τ	39
τ_{dead}	7

Table 5.6: Parameters for Ziegler Nichols tuning

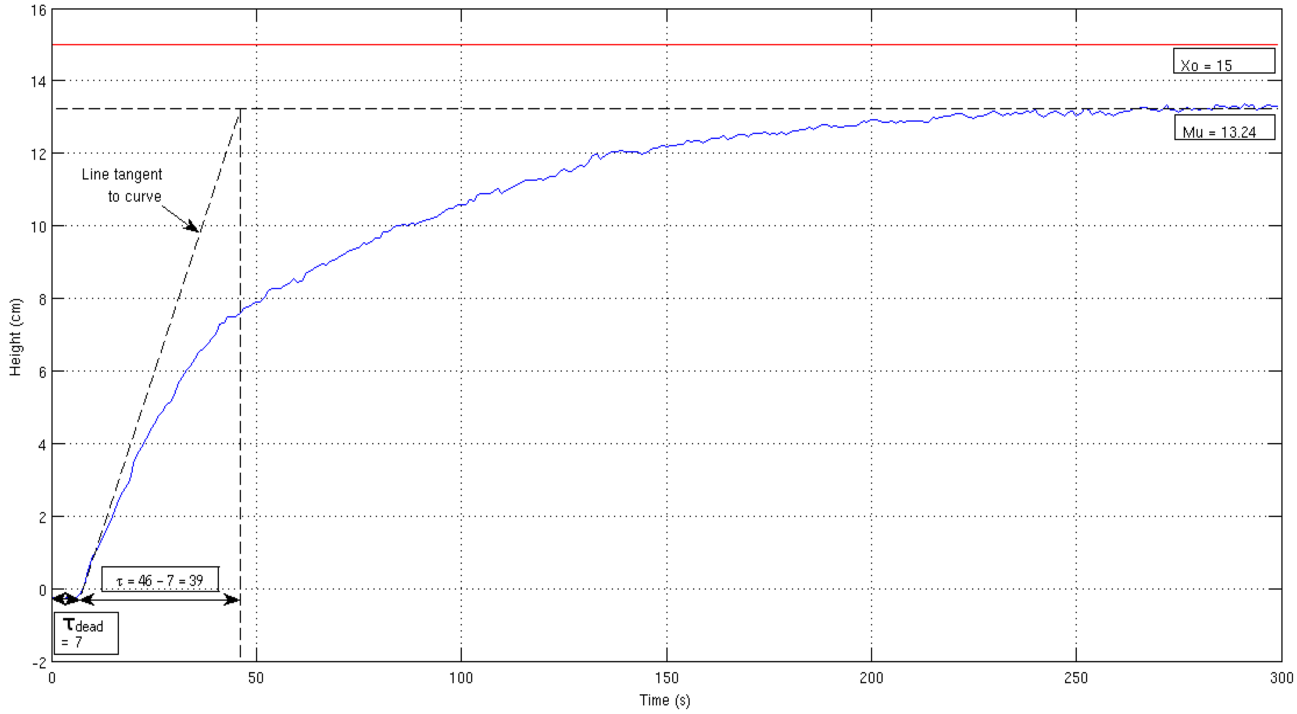

Figure 5.4: Extracting parameters to tune PID for set point 15cm

Using (31) we have

$$K_o = \frac{15 \times 39}{13.24 \times 7} = 6.312 \tag{32}$$

Using the formulas in table 5.5 we have gains (33)

$$\begin{gathered} K_c = 1.2\, K_o = 1.2 \times 6.312 = 7.5744 \\ T_i = 2\, \tau_{dead} = 2 \times 7 = 14 \\ T_d = 0.5\, \tau_{dead} = 0.5 \times 7 = 3.5 \end{gathered} \tag{33}$$

5.2.2.3 PID tuning for 25cm set point

Figure 5.5 shows parameter extraction necessary to implement Ziegler-Nichols rules. A tangent to the curve was made and then four parameters in total were calculated as shown in table 5.7

Parameter	Value
M_u	13.24
X_o	15
τ	39
τ_{dead}	7

Table 5.7: Parameters for Ziegler Nichols tuning

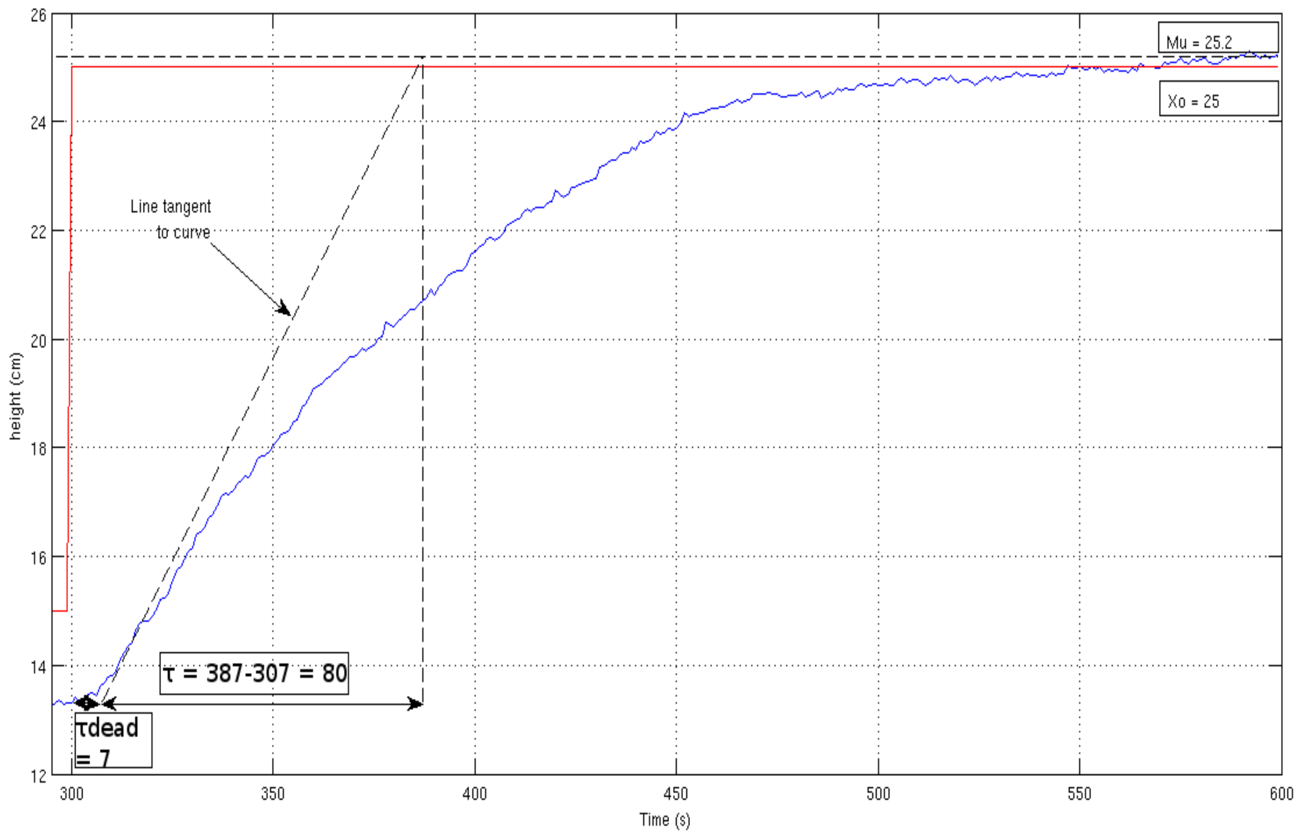

Figure 5.5: Parameter extraction for Ziegler-Nichols at 25cm

Using (31) we have

$$K_o = \frac{10 \times 80}{11.95 \times 7} = 9.564 \tag{34}$$

Using the formulas in table 5.5 we have gains (35)

$$\begin{gathered} K_c = 1.2\, K_o = 1.2 \times 9.564 = 11.477 \\ T_i = 2\, \tau_{dead} = 2 \times 7 = 14 \\ T_d = 0.5\, \tau_{dead} = 0.5 \times 7 = 3.5 \end{gathered} \tag{35}$$

Once the gains were calculated they were plugged in to the PID controller. The Simulink model of the plant is shown in figure 5.6.

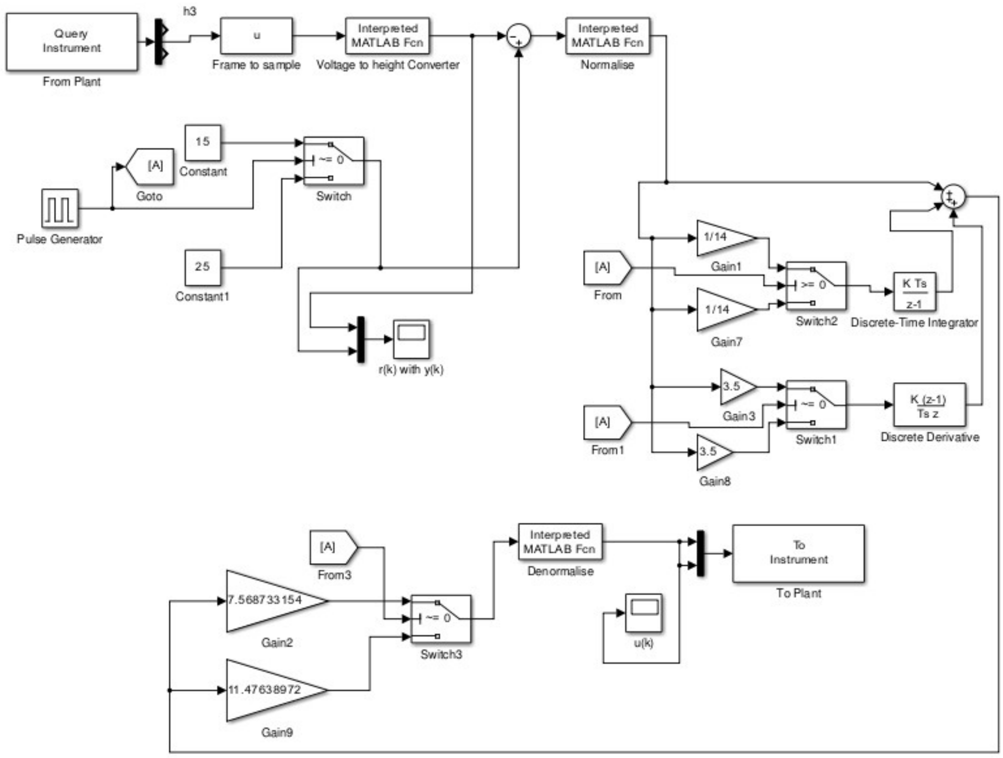

Figure 5.6: Simulink model of plant connected with PID controller.

The switches that can be seen in figure 5.6 are for gain scheduling. When the set point is changed so is the gain of the PID controller.

The response of the plant is shown in figure 5.7. PID controller has some overshoots which is the result of classic Ziegler-Nichols tuning method. It takes some time to reach the set point. Some disturbances were also added to the system such as pouring a glass of water in the tank. Its result can be seen in figure .

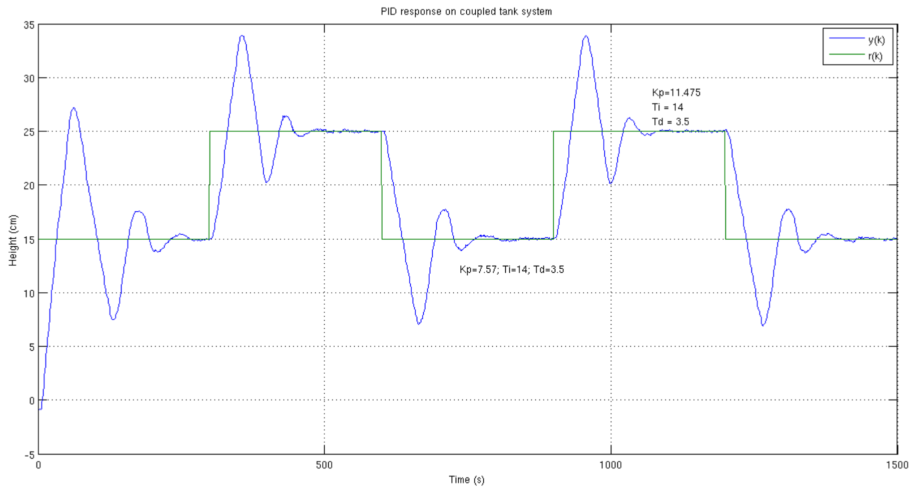

Figure 5.7: Response of plant on PID controller

5.2.3 PID response on disturbance

Figure 5.8 shows first the controller trying to maintain the height at set-point, 15cm. Then between 500 and 600 seconds water was poured in the tank 2 which can bee seen by a spike in the level of water in tank 2.

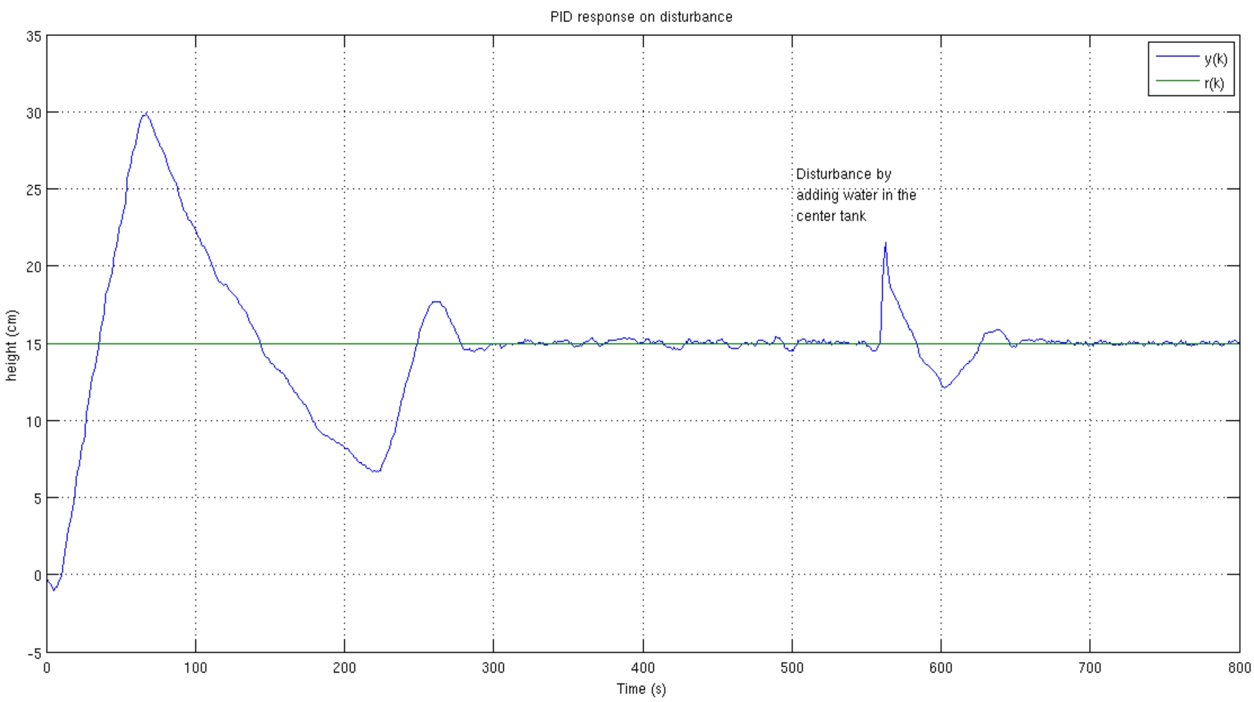

Figure 5.8: Response of PID on disturbance

Controller responded to it and then maintained the height at the set-point.

5.3. Adaptive fuzzy controller

5.3.1 Architecture of the fuzzy controllers

Architecture of the adaptive fuzzy control scheme is shown in figure 5.9. The large block represents the main fuzzy controller. This is a self organizing controller and it can only be tuned by the knowledge base present inside the controller. Looking further into the knowledge base, it can easily be seen that there are two major set of factors: the structure of controller, identified by the definition of the Membership Functions(MFs), and the rule consequents. Two auxiliary blocks now interact with he main controller namely: adaptation block(A – Block), and global learning block(GL – Block) to find appropriate parameters from evaluating our control architecture.

Regarding control signal as the output of the plant; using monotonicity coarse tuning of the rule consequents is done by A-Block. GL-Block is responsible for fine tuning of rule consequents, and MFs. Fine tuning, by GL-Block, is done by collecting I/O data from the progression of plant. When starting parameter values are not available i.e., in the preliminary iterations of control process, A-block contributes. After this Gl-Block takes over and fine tunes the control parameters.

Figure 5.9: Adaptive Controller Architecture

5.3.2 Test setup

Adaptive fuzzy controller was tested on coupled tank system, and it now controls the water level in tank 3 of the system. In this test we are not using global learning block and the controller has no information of the plant nor its is not tuned or pre-trained. The Simulink model of controller with plant interface is shown in figure 5.10.

The tank configuration is shown in figure 5.2.

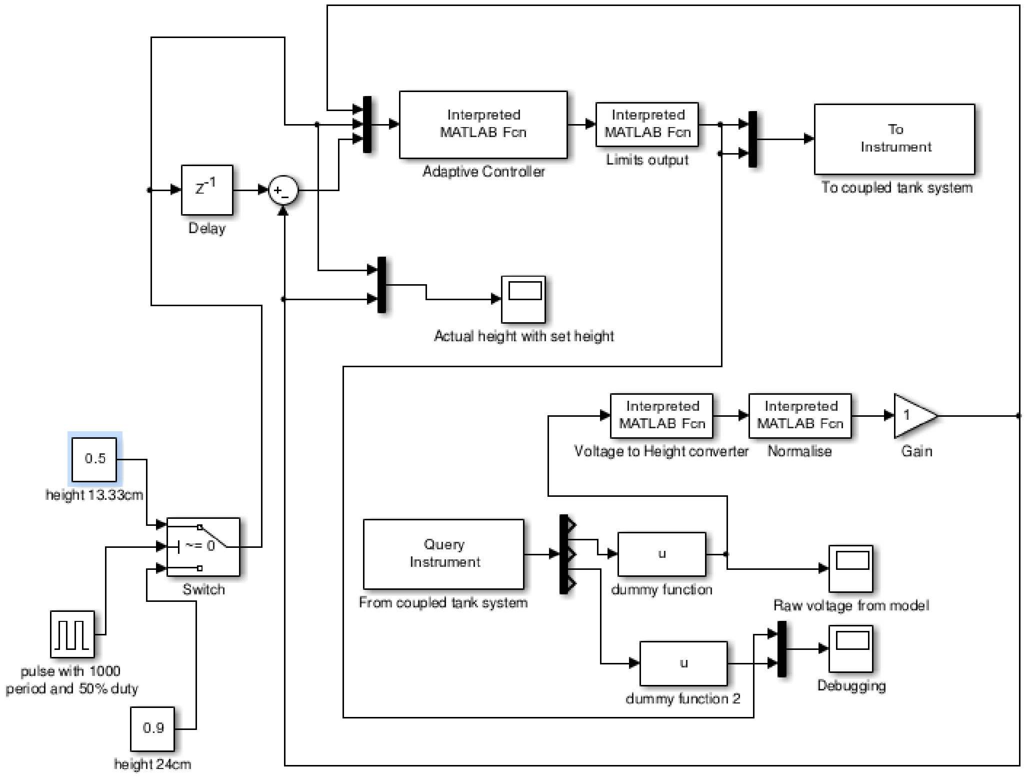

Figure 5.10: Simulink model of adaptive controller interfaced with coupled tank system

In this setup the adaptive controller was required to maintain the water level in tank 3 of the coupled tank system. The set point was a pulse train of 50% duty cycle, a period of 1000 seconds, amplitude of 0.2, and an offset by 0.5. This gives us two set-points at 0.5 and 0.7 with 500 seconds duration each. This can be seen in figure 5.11; blue line shows the set-points. 0.5 and 0.7 are normalized value of level; corresponds to 13.3 cm and 18.62 cm respectively. This setup gives us a water level in tank 3 at 13.3 cm for 500 seconds and then a level of 24 cm for another 500 seconds. This cycle is repeated again and we have two complete cycles.

Here, the controller is only controlling the flow rate of pump 1. It does this by sensing the height in tank 3 only; the controller has no information of the tank and does not monitors the flow rate between the tanks and out of the tank. It works on a goal oriented approach by sensing the current height, measuring the error, and then generating the required control signal.

Figure 5.10 depicts the Simulink model which contains the adaptive controller along with the interface blocks to control the plant and sense water level in tank. The MATLAB function “Adaptive Controller” contains the code for the adaptive fuzzy controller. Apart from that there are some helping MATLAB functions; Limit output, for saturating output when the input is beyond limits; Voltage to height converter, to convert voltage values to height in cm; and Normalize function, to normalize the height.

The results of the test are shown in figure 5.11. The plot is of actual height with respect to the set height.

Controller was evaluated for different C_0 and β, best results were achieved for $C_0 =$ 20 and β=100,000. In figure 5.11, an overshoot is seen due to inadequate learning of controller but once the controller gathers enough information about the plant it maintains the level as per the desired height. After 500 seconds, desired height changes and the controller subsequently changes the flow rate of the pump to maintain the new height. This goes on for another 500 seconds and then the cycle repeats.

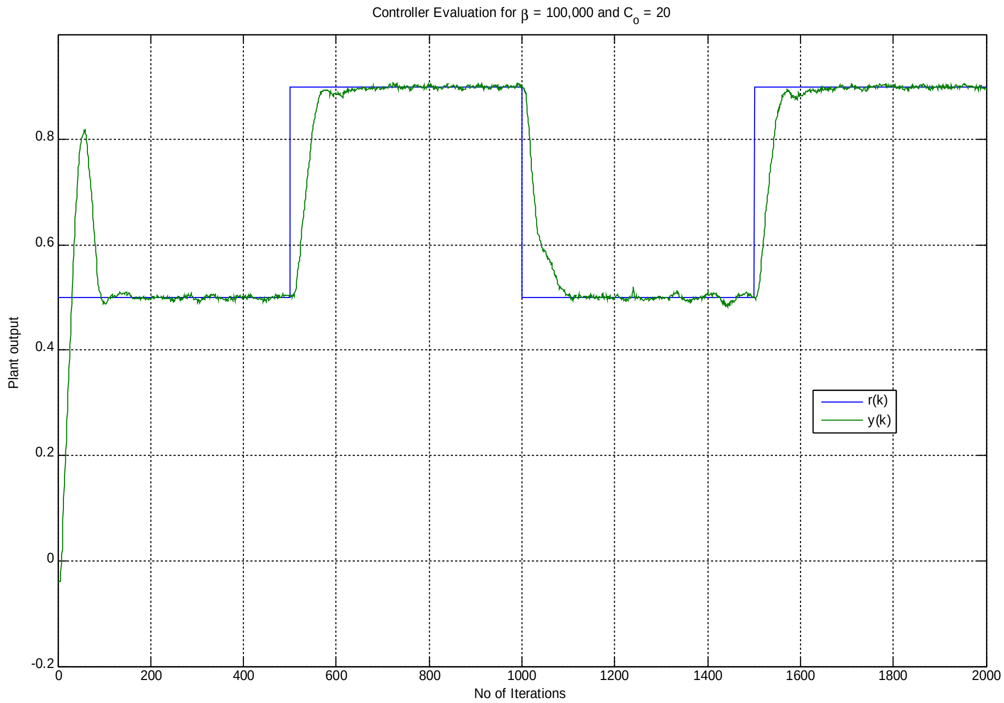

Figure 5.11: Controller evaluation of actual plant (Water level control of Three Tank System)

6 Conclusions and future recommendations

6.1. Conclusion

In this thesis a testing rig was designed, developed and tested for testing controllers . This testing rig is a coupled tank system due to its wide-spread usage in process industries such as beverage industry, food processing, effluent treatment, filtration, industrial chemical processing, water purification system, spray coating and pharmaceutical industries. This coupled tank system was designed such that it had three tanks which are coupled together with two pumps attached to left and right tanks. These pumps are the only source of liquid in the tanks. In addition every tank has an outlet for leakage of the liquid.

After tank fabrication, it needed level sensors for each tank. Four different level measuring techniques were used: capacitive sensing, ultrasonic transducers, IR distance sensors and pressure sensors. Of these four sensors, best results were obtained from pressure sensors.

After level sensing, pumps flow rate was made variable with the help of an inverter. This inverter drives the pump motor with different frequencies to give different flow rates.

The main controller generating pulses for inverter and acquiring data from level sensor is an ArduinoMega board. This board has an 8-bit microcontroller from Atmel and is programmed in Arduino wiring language. This board is interfaced with the computer running MATLAB, Simulink. This was done using Simulink Instrument Control Toolbox and serial interface.

Once tank was complete it was tested with a PID controller. PIDs gain were calculated using Ziegler-Nichols tuning rules at two different levels. Their results show satisfactory performance of tank.

Another controller, adaptive fuzzy controller, was also tested and showed improvement in performance over time. The set point in this case was also varied between two levels.

It was shown that with PID and adaptive fuzzy controller both were able to maintain liquid in the middle tank in SISO mode.

6.2. Future work recommendation

The following recommendations are suggested for further improvement in the testing rig.

- The testing rig can be improved by making it remotely accessible. This can be done by adding a camera for monitoring and connecting the coupled tank system's electronics to the Internet, enabling the tank to send and receive data via TCP/IP. Further the coupling valves and the leakage valves can also be automated so they can also be controlled remotely.
- The tank right now only connects to MATLAB or any other software that can work with serial ports. It can later be modified to accept any embedded system or small computer such as PLC or micro-controller so that the control system is developed on actual systems that are going to be used in the industries.

APPENDIX A (PROGRAM LISTING)

This appendix covers the source code for ArduinoMega2560. The Arduino programming language is an implementation of Wiring, a similar physical computing platform, which is based on the Processing multimedia programming environment. Following is the source code:

```
#include <avr/interrupt.h>
#include <avr/io.h>

#define INIT_TIMER_COUNT 156
#define RESET_TIMER2 TCNT2 = INIT_TIMER_COUNT
#define FREQ 50
#define MEGA2560


int value;
long timeOut;
String inputString = "";
boolean stringComplete = false;
int PWMA = 12;
int PWMB = 13;
byte FirstChannel = 0;
byte SecondChannel = 1;
byte ThirdChannel = 2;
long valueTimer=0;
int ledPin = 13;
byte out1a = 3;
byte out1b = 4;
byte out2a = 5;
byte out2b = 6;
boolean toggle = true;
int counter = 0;
int counter1 = 0;
int state = 0;
int counter2 = 0;
int state2 = 0;
short updateDEAD = 40;
short updateDEAD2 = 40;
short currentUP = 0;
short currentUP2 = 0;
short updateUP = 0;
short updateUP2 = 0;
short updatePWMcounter = 0;
unsigned long value1 = 0;
unsigned long value2 = 0;
unsigned long value3 = 0;
unsigned long SumValue1 = 0;
unsigned long SumValue2 = 0;
unsigned long SumValue3 = 0;
unsigned long valueNumber = 0;
byte received = 0;

short DEAD = 58;
short DEAD2 = 58;
```

```
int UP = 0;
int UP2 = 0;


short values[] =
{1,263,261,259,258,256,254,253,251,250,248,247,245,244,242,241,239,23
8,237,235,234,233,231,230,229,227,226,225,224,222,221,220,219,218,216
,215,214,213,212,211,210,209,208,207,206,205,204,203,202,201,200,199,
198,197,196,195,194,193,192,191,190,189,188,188,187,186,185,184,183,1
83,182,181,180,179,179,178,177,176,175,175,174,173,173,172,171,170,17
0,169,168,168,167,166,166,165,164,164,163,162,162,161};
short valuesDead[] =
{200,26,26,26,26,26,26,25,25,25,25,25,25,24,25,24,24,24,24,24,24,23,2
3,23,23,23,23,23,22,23,22,22,22,22,22,22,22,22,21,21,21,21,21,21,21,2
0,20,20,20,20,20,20,20,20,20,20,20,19,19,19,19,19,19,19,19,19,19,19,1
9,18,18,18,18,18,18,18,18,18,18,18,18,18,17,17,17,17,17,17,17,17,17,1
7,17,17,17,16,16,17,16,16};
short values2[] =
{1,175,175,174,173,173,172,172,171,171,170,170,169,169,168,168,167,16
7,166,166,165,165,165,164,164,163,163,162,162,161,161,160,160,160,159
,159,158,158,157,157,156,156,156,155,155,154,154,154,153,153,152,152,
152,151,151,150,150,150,149,149,148,148,148,147,147,147,146,146,145,1
45,145,144,144,144,143,143,143,142,142,141,141,141,140,140,140,139,13
9,139,138,138,138,137,137,137,136,136,136,136,135,135};
short valuesDead2[] =
{200,18,17,18,18,17,18,17,18,17,17,17,17,17,17,17,17,17,17,17,17,17,1
6,17,16,17,16,17,16,16,16,16,16,16,16,16,16,16,16,16,16,16,16,16,16,1
6,16,15,16,15,16,15,15,15,15,15,15,15,15,15,15,15,15,15,15,15,15,15,1
5,15,15,15,15,14,15,14,14,15,14,15,14,14,14,14,14,14,14,14,14,14,14,1
4,14,14,14,14,14,14,14,14};

ISR(TIMER2_OVF_vect) {
//timer overflows every 50uS
  RESET_TIMER2;
  updatePWMcounter++;
  switch(state){
    case 0: //dead zone
    if( ++counter1 == DEAD ){
      counter1 = 0;
      state = 1;
      digitalWrite(out1a, LOW);
    }
    break;
    case 1: //high zone
    if( ++counter1 == UP ){
      counter1 = 0;
      state = 2;
      digitalWrite(out1a, HIGH);
      digitalWrite(out1b, HIGH);
    }
    break;
    case 2: //deadzonse
    if( ++counter1 == DEAD ){
      counter1 = 0;
      state = 3;
    }
    break;
    case 3: //deadzone
    if( ++counter1 == DEAD ){
      counter1 = 0;
      state = 4;
```

```
    digitalWrite(out1b, LOW);
  }
  break;
  case 4: //low zone
  if( ++counter1 == UP ){
    counter1 = 0;
    state = 5;
    digitalWrite(out1a, HIGH);
    digitalWrite(out1b, HIGH);
  }
  break;
  case 5: //deadzone
  if( ++counter1 == DEAD ){
    counter1 = 0;
    state = 0;
    UP = values[currentUP];
    DEAD = valuesDead[currentUP];
  }
  break;
  default: break;
}
  switch(state2){
  case 0: //dead zone
  if( ++counter2 == DEAD2 ){
    counter2 = 0;
    state2 = 1;
    digitalWrite(out2a, LOW);
  }
  break;
  case 1: //high zone
  if( ++counter2 == UP2 ){
    counter2 = 0;
    state2 = 2;
    digitalWrite(out2a, HIGH);
    digitalWrite(out2b, HIGH);
  }
  break;
  case 2: //deadzonse
  if( ++counter2 == DEAD2 ){
    counter2 = 0;
    state2 = 3;
  }
  break;
  case 3: //deadzone
  if( ++counter2 == DEAD2 ){
    counter2 = 0;
    state2 = 4;
    digitalWrite(out2b, LOW);
  }
   break;
  case 4: //low zone
  if( ++counter2 == UP2 ){
    counter2 = 0;
    state2 = 5;
    digitalWrite(out2a, HIGH);
    digitalWrite(out2b, HIGH);
  }
  break;
  case 5: //deadzone
  if( ++counter2 == DEAD2 ){
    counter2 = 0;
```

```
      state2 = 0;
      UP2 = values2[currentUP2];
      DEAD2 = valuesDead2[currentUP2];
    }
    break;
  }

};

void setup(){
  pinMode(ledPin, OUTPUT);
  digitalWrite(out1a, HIGH);
  digitalWrite(out1b, HIGH);
  digitalWrite(out2a, HIGH);
  digitalWrite(out2b, HIGH);
  pinMode(out1a, OUTPUT);
  pinMode(out1b, OUTPUT);
  pinMode(out2a, OUTPUT);
  pinMode(out2b, OUTPUT);

  ADCSRA &= ~(0x01);

  Serial.begin(57600);
  analogReference(INTERNAL2V56);
  SumValue1 = 0;
  SumValue2 = 0;
  SumValue3 = 0;
  valueTimer = millis() + 4000;
  UP = values[0];
  UP2 = values[0];
  updateUP = 0;
  currentUP = 0;
  updateUP2 = 0;
  currentUP2 = 0;
  counter1 = 0;
  counter2 = 0;
  state = 0;
  state2 = 0;
  TCCR2A = 0;
  TCCR2B = 0;
//  Prescaler set to divide by 8
  TCCR2B |= ((0<<CS22) | (1<<CS21) | (0<<CS20));

  // Use normal mode
  //TCCR2 |= (0<<WGM21) | (0<<WGM20);
  // Use internal clock - external clock not used in Arduino
  ASSR &= ~(0<<AS2);
  TIMSK2 = 0;
  TIMSK2 |= (1<<TOIE2);        //Timer2 Overflow Interrupt Enable
  RESET_TIMER2;
  sei();
  delay(2000);
}

void loop(){
  communication();
  transmitValues();
  updatePWM();
}

void updatePWM(){
```

```
  if( updatePWMcounter > 399 ){
    updatePWMcounter = 0;
    if( currentUP < updateUP ){
      if (currentUP < 99)
        currentUP++;
    }
    else if(currentUP > updateUP) {
      if( currentUP > 0 )
        currentUP--;
    }

    if( currentUP2 < updateUP2 ){
      if( currentUP2 < 99 )
        currentUP2++;
    }
    else if(currentUP2 > updateUP2 ){
      if ( currentUP2 > 0 )
        currentUP2--;
    }
  }
}

void serialEvent() {
  while (Serial.available()) {
    // get the new byte:
    char inChar = (char)Serial.read();
    // add it to the inputString:
    inputString += inChar;
    // if the incoming character is a newline, set a flag
    // so the communication function knows there is data
    timeOut = millis() + 10;
    if (inChar == '\n') {
      toggle = toggle ^ true;
      stringComplete = true;
    }
  }
}


void communication(){
  if( millis() > timeOut ){
    inputString = "";
  }
  if( stringComplete ){
    char temp[5];
    if( inputString.indexOf ("MOT") == 0){

      int start = inputString.indexOf("(") + 1;
      int endI = inputString.indexOf(",");

      inputString.substring(start,endI).toCharArray(temp,4);
      value = atoi(temp);
      updateUP = value;
      start = endI+1;
      endI = inputString.indexOf(")",start);
      inputString.substring(start,endI).toCharArray(temp,4);
      value = atoi(temp);
      updateUP2 = value;
    }
    else if( inputString.indexOf ("DEAD") == 0 ){
      int start = inputString.indexOf("(") + 1;
```

```
      int endI = inputString.indexOf(")");
      inputString.substring(start,endI).toCharArray(temp,4);
      updateDEAD = atoi(temp);
    }else if( inputString.indexOf ("PRINT") == 0 ){
      Serial.println("UP = " + String(UP) + " UP2 = " +String(UP2)+ "
DEAD = " + String(DEAD) + " DEAD2 = " + String(DEAD2));
    }
    inputString = "";
    stringComplete = false;
  }
}

void transmitValues(){
  if ( millis() > valueTimer ){
    valueTimer = millis() + 1000;
    Serial.println(String(value1) + "," +
String(value2)+","+String(value3)+","+currentUP2 + "," + UP2);
  }
  if ( counter > 4095){
    counter = 0;
    valueNumber ++;
    value1 = SumValue1;
    value2 = SumValue2;
    value3 = SumValue3;
    SumValue1 = 0;
    SumValue2 = 0;
    SumValue3 = 0;
  }
  SumValue1 += analogRead(0);
  SumValue2 += analogRead(1);
  SumValue3 += analogRead(2);
  counter++;
}
```

REFERENCES

1. F.G. Shinskey, “Averaging level control,” Chemical Processing, vol.60, no.9, Sep 1997, pp. 58-58.
2. H. Pana, H. Wonga, V. Kapilaa, M.S. de Queiroz, “Experimental validation of a nonlinear backstepping liquid level controller for a state coupled two tank system”, Control Engineering Practice, vol. 13, no. 1, Jan. 2005, pp.27-40
3. Visioli, “A new design for a PID plus feed-forward controller. J. Process Control,” 2004, pp. 457-463.
4. N. Guigues, C. Berho, J.-C. Foucher, A.-M. Fouillac, “The use of tank experiments in assessing the performance of 7 emerging tools in optimizing water monitoring,” Trends in Analytical Chemistry, vol. 26, no. 4, April 2007, pp. 268-273.
5. Rajni Jain, T.Vinopraba, N.Sivakumaran,T.K.Radhakrishnan3, “Design and implementation of controllers for MIMO process ,” 2009, DOI 10.1109/ARTCom.2009.152 .
6. Paul A. Jensen, “Operations Research models and methods,” 2004, pp 1 - 2
7. C. C. Ko, Senior Member, IEEE, Ben M. Chen, member, ieee, jianping chen, yuan zhuang, and kay chen tan, member, ieee “development of a web-based laboratory for control experiments on a coupled tank apparatus,” ieee transactions on education, vol. 44, no. 1, february 2001.
8. M. Shor and A. Bhandari, “Access to an instructional control laboratory experiment through the World Wide Web,” in Proc. 1998 Amer. Contr. Conf., Philadelphia, PA, 1998, pp. 1319–1325.
9. Adolfo BAUCHSPIESS , Bruno GUIMARÃES , Hugo L. GOSMANN , “Remote Experimentation on a Three Coupled Water Reservoirs ,”, 2003.
10. Ivan Holič and Vojtech Veselý, “Robust PID controller design for coupled-tank process”, 18th International Conference on Process Control june 14-17,2011.
11. Varsha Bhambhani and YangQuan Chen , “Experimental Study of Fractional Order Proportional Integral (FOPI) Controller for Water Level Contr ol,” Proceedings of the 47th IEEE Conference on Decision and Control , Dec 9 – 11, 2008.
12. Anouar Benamor, Larbi Chrifi-alaui, Hassani Messaoud, Mohamed Chaabane, “Sliding Mode Control, with Integrator, for a Class of MIMO Nonlinear Systems ,” Scientific Research, Engineering, 2011, 3, 435-444.
13. Arjin Nurnsomrarr, Tianchai Suksri, and Maitree Thumma, “Design of 2-DOF PI Controller with Decoupling for Coupled-Tank Process ,” International Conference on Control , Automation and Systems 2007 , Oct 17-20, 2007.
14. Actel application note, “Improving ADC results through oversampling and post-processing of data”, pp 5 – 14.
15. Atmel application note, “Enhancing ADC resolution by oversampling”, pp 3 - 10.

REFERENCES

16. J.G. Ziegler and N.B. Nichols, “Optimum settings for automatic controllers,” Trans. ASME, vol. 64, pp. 759–768, Nov. 1942
17. Astrom, Karl J., Hagglund, Tore.,”PID control theory designing and tuning”, 2nd Edition, pp 134-146.
18. IEEE Control Systems Magazine, February 2006, p 93.