

**NEURO-FUZZY LOGIC CONTROLLER FOR HEAT EXCHANGER  
TEMPERATURE CONTROL**

by

**MOHD ADZRIL BIN ABDUL RAZAK**

**FINAL PROJECT REPORT**

**Submitted to the Electrical & Electronics Engineering Programme  
in Partial Fulfillment of the Requirements  
for the Degree  
Bachelor of Engineering (Hons)  
(Electrical & Electronics Engineering)**

**Universiti Teknologi Petronas  
Bandar Seri Iskandar  
31750 Tronoh  
Perak Darul Ridzuan**

**© Copyright 2007  
by  
Mohd Adzril bin Abdul Razak, 2007**

# **CERTIFICATION OF APPROVAL**


## **NEURO-FUZZY LOGIC CONTROLLER FOR HEAT EXCHANGER TEMPERATURE CONTROL**

by

**Mohd Adzril bin Abdul Razak**

A project dissertation submitted to the  
Electrical & Electronics Engineering Programme  
Universiti Teknologi PETRONAS  
in partial fulfilment of the requirement for the  
Bachelor of Engineering (Hons)  
(Electrical & Electronics Engineering)

Approved:

  
\_\_\_\_\_  
Ms Suhaila Badarol Hisham  
Project Supervisor

**UNIVERSITI TEKNOLOGI PETRONAS  
TRONOH, PERAK**

June 2007

## **CERTIFICATION OF ORIGINALITY**

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.



---

**Mohd Adzril bin Abdul Razak**

## ABSTRACT

A commonly used controller in the process industries is the Proportional-Integral-Derivative (PID) controller due to its features; cheap and easy to configured. Another type of controller that is now being developed is neuro-fuzzy logic controller that functions like a human brain which consists of interconnected processing elements called nodes or neurons that work together to produce an output function. The feature that it has that the PID controller does not have is the ability to be retrained to deal with various conditions in the process industries. A previous final year project on neuro-fuzzy logic controller yielded an unsatisfactory result as it could only perform for a single set point change [1]. The objective of this project is to improve the neuro-fuzzy logic controller so that it can control a process for a wider range of set point values. Data achieved for this project are through plant experiments using SIM 305 Pilot Plant: Plant 6 for the purpose of process modeling and computer simulation. Computer simulation is used to design the PID controller and the neuro-fuzzy logic controller. These two types of controllers are then compared and analyzed based on their performances. Controlled Variable (CV) Overshoot, Manipulated Variable (MV) Overshoot and Decay Ratio are the benchmarks used to compare and evaluate these two controllers. Based on the simulation results, the two controllers are in par since the benchmark values for the two controllers are nearly the same. However, it can be concluded at this stage that the PID controller is better than the neuro-fuzzy logic controller due to the smallest value of overshoot compared to the neuro-fuzzy logic controller.

## **ACKNOWLEDGEMENTS**

First of all, my greatest thanks to Allah s.w.t. for His bless giving me the chance to finish my Final Year Project.

I would like to express my gratitude to my FYP supervisor, Ms. Suhaila Badarol Hisham upon close supervision and guidance throughout the project commencement. All valuable knowledge taught, project advised, supervised; kindness and cooperation given were really appreciated

Thank you to Mr. Azhar Zainal Abidin, UTP Pilot Plant Supervisor, for full co-operation and commitment as well as guidance while performing the plant experiments for the data acquisition phase of the project. Thank you as well to Mrs. Siti Hawa Hj. Mohd Tahir, EE Lab Technician, for all the information and assistant given during the whole year project commencement.

Special thanks to Muhammad Faizal Ja'afar and Mohd Rozairi, ex-UTP students for their report and finding that is used as guidance towards completion of this project.

Last but not least, thanks to my family and my colleague for giving me hope and courage when I need it the most.

## TABLE OF CONTENTS

LIST OF TABLES .....	x
LIST OF FIGURES .....	xi
LIST OF ABBREVIATIONS .....	xiii
<b>CHAPTER 1 INTRODUCTION .....</b>	<b>1</b>
1.1 Background of Study.....	1
1.2 Problem Identification.....	2
1.3 Objectives.....	2
1.4 Scope of the Project.....	3
1.4.1 Gather Experimental Data .....	3
1.4.2 Empirical System Modeling .....	3
1.4.3 Designing the Controller.....	3
1.4.4 Train the Neuro-Fuzzy Logic Controller with Various Set Point Change.....	3
1.4.5 Analysis and Comparison .....	4
<b>CHAPTER 2 LITERATURE REVIEW .....</b>	<b>5</b>
2.1 Heat Exchanger .....	5
2.2 Fuzzy logic .....	6
2.3 Neural Networks.....	6
2.4 Proportional-Integral-Derivative (PID) Controller .....	6
2.5 Empirical Model Identification .....	9
2.5.1 Experimental Design .....	9
2.5.2 Plant Experiment .....	10
2.5.3 Determining Model Structure .....	10
2.5.4 Parameter Estimation.....	10
2.5.5 Diagnostic Evaluation.....	10
2.5.6 Verification .....	10
2.6 The Process Reaction Curve.....	11
2.6.1 Method I.....	12
2.6.2 Method II .....	12

2.7 Ziegler-Nichols Open-Loop Tuning.....	13
2.8 Ziegler-Nichols Closed-Loop Tuning Correlation.....	13
2.9 Feedforward Controller.....	13
2.10 ANFIS.....	15
2.11 Previous Work.....	15
<b>CHAPTER 3 METHODOLOGY/ PROJECT WORK.....</b>	<b>16</b>
3.1 Procedure and Identification.....	16
3.2 Tools and Equipments.....	17
3.3 Data Gathering.....	17
3.4 Data Processed.....	18
3.5 PID Tuning.....	18
3.6 Disturbance Model and Feedforward Controller.....	19
3.7 Simulation Block and Testing.....	22
3.7.1 The Decay Ratio.....	24
3.7.2 CV Overshoot.....	24
3.7.3 MV Overshoot.....	25
3.8 Generating Neuro-Fuzzy Logic.....	26
3.8.1 Data Import to ANFIS.....	27
3.8.2 Train the Neuro-Fuzzy Logic Controller.....	28
3.8.3 Creating the .fis File.....	29
3.8.2 Load the Trained Neuro-Fuzzy Logic Controller.....	29
3.9 Design Neuro-Fuzzy Logic Controller to Accept Multiple Training Data.....	30
3.9.1 Alternative 1.....	27
3.9.2 Alternative 2.....	28
<b>CHAPTER 4 RESULT AND DISCUSSION.....</b>	<b>33</b>
4.1 The Process Transfer Function.....	33
4.2 Testing the PID Controller.....	33
4.3 Fine Tuning.....	36
4.4 Feedforward Controller.....	38

4.5 Training Feedback Neuro-Fuzzy Logic Controller .....	41
4.5.1 Alternative 1 vs Alternative 2.....	41
4.5.2 Alternative 2 to Train the Controller .....	41
4.6 Performance Check and Comparison .....	43
<b>CHAPTER 5 CONCLUSION AND RECOMMENDATION .....</b>	<b>46</b>
5.1 Conclusion.....	46
5.2 Recommendation.....	47
REFERENCES.....	48
APPENDICES .....	49
APPENDIX A OVERVIEW OF SIM305 PILOT PLANT .....	49
APPENDIX B RATE OF DIFFERENCE.....	50
APPENDIX C SIMULATION BLOCK DIAGRAM.....	55



## LIST OF TABLES

Table 2.1	Parameters to be calculated using Method I .....	12
Table 2.2	Parameters to be calculated using Method II .....	12
Table 3.1	Ziegler-Nichols open-loop tuning.....	19
Table 3.2	Ziegler-Nichols closed-loop tuning correlation .....	19
Table 3.3	Evaluation of Potential Feedforward Variables .....	20
Table 4.1	Calculated variables for PID using Ziegler-Nichols Open-loop tuning.... .....	34
Table 4.2	Performance check using Z-N Open-loop tuning .....	34
Table 4.3	Calculated variables for PID using Ziegler-Nichols Closed-loop tuning correlation .....	35
Table 4.4	Performance check using Z-N Closed-loop tuning correlation .....	35
Table 4.5	The initial values to be used for fine tuning.....	36
Table 4.6	The values obtained after the fine tune .....	36
Table 4.7	The finalized values obtained after the fine tune with addition of $T_D$ ..	37
Table 4.8	Performance check with step change of 23% of valve opening .....	43
Table 4.9	Performance check with step change of 15% of valve opening .....	44
Table 4.10	Data used to train neuro-fuzzy logic controller for feedforward controller .....	45

## LIST OF FIGURES

Figure 2.1	Shell-and-tube-heat exchanger with one shell pass and one tube pass; cross-counterflow operation.....	5
Figure 2.2	Example of control loop .....	7
Figure 2.3	Empirical Model Building Procedure.....	9
Figure 2.4	Ideal Process Reaction Curve based on the assumption of the output is first-order-with-dead-time.....	11
Figure 2.5	Graphical approach for Method I .....	12
Figure 2.6	Graphical approach for Method II .....	12
Figure 2.7	Simplified block diagram of feedforward compensation .....	14
Figure 3.1	Method used to accomplish the objective of the project .....	16
Figure 3.2	Result obtained from plant experiment .....	17
Figure 3.3	Result obtained from the previous final year project.....	18
Figure 3.4	Feedback-Only Control Performance.....	21
Figure 3.5	Load Disturbance Reaction Curve Plotted from Data .....	21
Figure 3.6	The control process block diagram using a single block PID controller.. .....	22
Figure 3.7	Similar control process block diagram but using different PID controller .....	22
Figure 3.8	Typical CV response of a feedback control system to a step SP change . .....	23
Figure 3.9	Typical MV response of a feedback control system to a step SP change. .....	25
Figure 3.10	An additional block added for the purpose of importing data to the workspace.....	26
Figure 3.11	Simplified methodology to implement the neuro-fuzzy logic controller .....	26
Figure 3.12	Data to be used for training the neuro-fuzzy logic controller.....	27
Figure 3.13	The plotted MV as it is loaded into the ANFIS toolbox.....	27
Figure 3.14	The mapping structure obtained by generating the FIS.....	28
Figure 3.15	The plotted training result reduces error close to zero .....	28

Figure 3.16	The neuro-fuzzy logic relationship between the input and output based on Sugeno type.....	29
Figure 3.17	A new process block diagram where PID is replaced with fuzzy-logic controller .....	30
Figure 3.18	Methodology for Alternative 1 .....	31
Figure 3.19	Methodology for Alternative 2 .....	32
Figure 4.1	The transient response of the feedback control system to a step set point change (using Z-N Open-Loop Tuning) .....	34
Figure 4.2	The transient response of the feedback control system to a step set point change (using Z-N Closed-Loop Tuning Correlation).....	35
Figure 4.3	Tuning using PI-mode .....	37
Figure 4.4	Tuning using PID-mode .....	37
Figure 4.5	The complete process block diagram .....	39
Figure 4.6	The effect of disturbance to the process .....	40
Figure 4.7	The effect of disturbance to the process is reduced by the feedforward controller .....	40
Figure 4.8	How the training set is input into the ANFIS toolbox .....	42
Figure 5.1	Proposed control strategy with two controllers working together.....	47

## LIST OF ABBREVIATIONS

<b>ANFIS</b>	Adaptive Neuro-Fuzzy Inference System
<b>MV</b>	Manipulated Variable
<b>PID</b>	Proportional, Integral and Derivative
<b>CV</b>	Process, or Controlled Variable
<b>SP</b>	Set Point
<b>T<sub>s</sub></b>	Settling Time; the time required for the CV to reach and stay within $\pm 2\%$ of the steady-state value
<b>OS</b>	Percentage Overshoot
<b>DR</b>	Decay ratio
<b>CV OS</b>	Controlled Variable Percentage Overshoot
<b>MV OS</b>	Manipulated Variable Percentage Overshoot

# **CHAPTER 1**

## **INTRODUCTION**

### **1.1 Background of Study**

Process control is a statistics and engineering discipline that deals with architectures, mechanisms, and algorithms for controlling the output of a specific process. This process control is important in order to maintain variable at the desired value when disturbance occurs and to respond to the changes in the desired value or set point.

Proportional, Integral, and Derivative (PID) controller is a common feedback loop component in process control systems. It operates by taking a measured value from a process and compares it with a reference set point value. The difference or error signal is then used to adjust the input to the process in order to bring the value of process' measured back to its desired set point.

Another intelligent type of controller, neuro-fuzzy logic controller that is used for the same purpose operates in a different ways. This controller is modeled like the human brain which consists of interconnected processing elements called nodes or neurons that work together to produce an output function. The output relies on the cooperation of the individual neurons within the network to operate. Basically each neuron contains information specified by the user. Processing of information by these networks is done in parallel rather than in series. Therefore, for this controller to operate effectively, it needs to be trained with a multiple training set of data or information for the networks to establish relationship between these neurons and produce the desired output.

## **1.2 Problem Identification**

The previous final year project demonstrates a major flaw on the controller as it could only perform for a single set point change. The controller could only operate for a single set point change of 0% to 20% opening of the valve. When the set point is changed to other values, it yields an undesired result. This needs to be overcome so that the controller can perform for the whole range of the process.

The previous controller was found to have insufficient set of data for the training purpose of the neuro-fuzzy logic controller. Therefore, several sets of data are to be collected and used to train the neuro-fuzzy logic controller so that it could perform the required task.

## **1.3 Objectives**

The main objective of this project is to improve the performance of the neuro-fuzzy logic controller built in the previous final year project. In order to improve this controller, several set of data are gathered for the purpose of modeling the PID controller and training the neuro-fuzzy logic controller.

The performance of these two controllers, the neuro-fuzzy logic controller and the PID controller will be compared based on several benchmarks which are the decay ratio, manipulated variable (MV) overshoot and controlled variable (CV) overshoot. The project will also try to identify the advantages and the limitations of using neuro-fuzzy logic controller for process control applications.

## **1.4 Scope of the Project**

The project will focus on the feedforward-feedback temperature control of a heat exchanger. The scopes of the project are as below:

### **1.4.1 Gather Experimental Data**

Plant experiment is conducted in order to collect the required data for the purpose of system modeling. SIM 305 Pilot Plant: Plant 6 (refer to Appendix A) is used for the experiment as it consist of a feedforward-feedback temperature control of the heat exchanger.

### **1.4.2 Empirical System Modeling**

The models are determined by making changes in the input variable during the plant experiment. The resulting output or dynamic response is used to estimate the model parameter. This empirical modeling is an important procedure to develop an effective controller.

### **1.4.3 Designing the Controller**

The parameters obtained from the system modeling are used to design the PID controller. This controller is design using MATLAB Simulink.

### **1.4.4 Train the Neuro-Fuzzy Logic Controller with Various Set Point Change**

The error and the output or controlled variable (CV) resulting from the performance of the PID controller designed will be used as the multiple data set to train the neuro-fuzzy logic controller.

#### **1.4.5 Analysis and Comparison**

These controllers, the PID controller and the neuro-fuzzy logic controller will be analyzed and compared based on their performance. This process enables identification of the advantages and limitations of using neuro-fuzzy logic controller in process control applications.



## CHAPTER 2

### LITERATURE REVIEW

#### 2.1 Heat Exchanger

Heat exchanger is a device that enables heat to be transferred from one fluid to another. This transfer process can be done in two ways, (i) whether the fluids are separated by a solid wall so that they never mix, or (ii) the fluids are directly contacted.

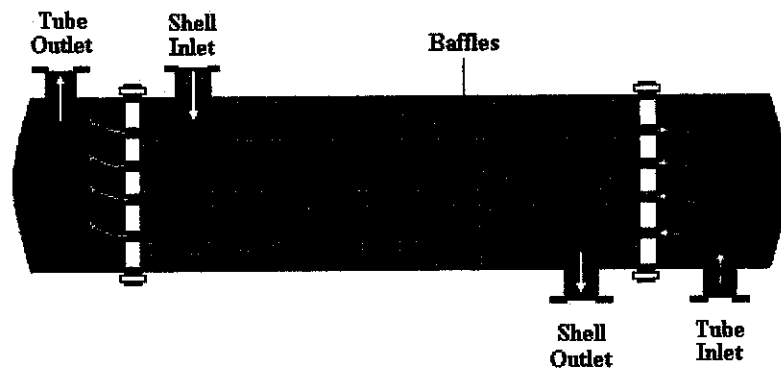


Figure 2.1: Shell-and-tube-heat exchanger with one shell pass and one tube pass; cross-counterflow operation. [5]

Two fluids, of different starting temperatures, flow through the heat exchanger. One flows through the tubes (the tube inlet) and the other flows outside the tubes but inside the shell (the shell inlet). Heat is transferred from one fluid to the other through the tube walls, either from tube side to shell side or vice versa. In order to transfer heat efficiently, a large heat transfer area should be used, so there are many tubes. In this way, waste heat can be put to use and a great way to conserve energy. [5]

## **2.2 Fuzzy logic**

Fuzzy logic is derived from fuzzy set theory dealing with reasoning that is approximate rather than precisely deduced from classical predicate logic. It can be thought of as the application side of fuzzy set theory dealing with well thought out real world expert values for a complex problem. [6]

## **2.3 Neural Network**

Neural networks are an interconnected assembly of simple processing elements, units or nodes, whose functionality is loosely based on the animal brain. The processing ability of the network is stored in the inter-unit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns. Neural nets are used in bioinformatics to map data and make predictions. [7]

## **2.4 Proportional-Integral-Derivative (PID) Controller**

Proportional-Integral-Derivate controller or PID controller is a common feedback loop used in industries.

Advantages of this controller:-

- Can adjust the process output based on the historical data unlike a simple control algorithm.
- Can also adjust the rate of the change of the error signal and this gives it more accurate and stable control.
- Does not require advanced mathematics to design and can be easily adjusted as per required.
- Can be used to control any measurable variable which can be affected by manipulating some other process variable

The controller has a reference Set Point (SP) in which the SP will be used to compare with the measured value which may come from any process. The difference then is used to calculate the new input in order for the process to return to normal or desired measurement.

The basic loop of this controller consists of three parts; the first part will be the measurement by a sensor connected to the process, then the decision in the controller element and lastly, action through an output devices.

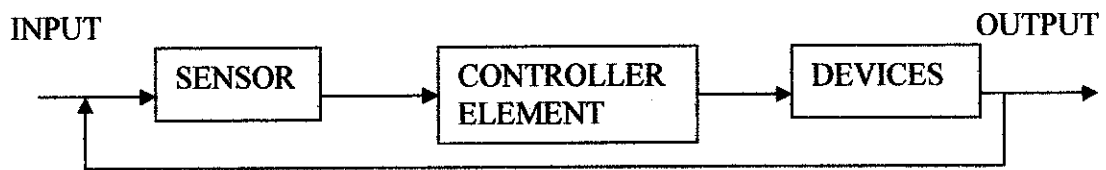


Figure 2.2: Example of control loop

Theoretically, PID is named after its correcting calculation;

*Proportional* - To handle the present, the error is multiplied by a (negative) constant P (for "proportional"), and added to (subtracting error from) the controlled quantity. P is only valid in the band over which a controller's output is proportional to the error of the system.

*Integral* - To handle the past, the error is integrated (added up) over a period of time, and then multiplied by a (negative) constant I (making an average), and added to (subtracting error from) the controlled quantity. I average the measured error to find the process output's average error from the SP. A simple proportional system oscillates, moving back and forth around the SP, because there's nothing to remove the error when it overshoots. By adding a negative proportion of the average error from the process input, the average difference between the process output and the SP is always being

reduced. Therefore, eventually, a well-tuned PID loop's process output will settle down at the SP.

*Derivative* - To handle the future, the first derivative (the slope of the error) over time is calculated, and multiplied by another (negative) constant D, and also added to (subtracting error from) the controlled quantity. The derivative term controls the response to a change in the system. The larger the derivative term, the more rapidly the controller responds to changes in the process's output. Its D term is the reason a PID loop is also called a "Predictive Controller." The D term is reduced when trying to dampen a controller's response to short term changes. Practical controllers for slow processes can even do without D.

## 2.5 Empirical Model Identification

Empirical modeling method provides dynamic relationship between selected input and output variables from experimental data. The models are determined by making small changes in the input variable(s) about nominal operating condition and the resulting dynamic response is used to determine the model. Empirical Model building procedure is as in Figure 2.3.

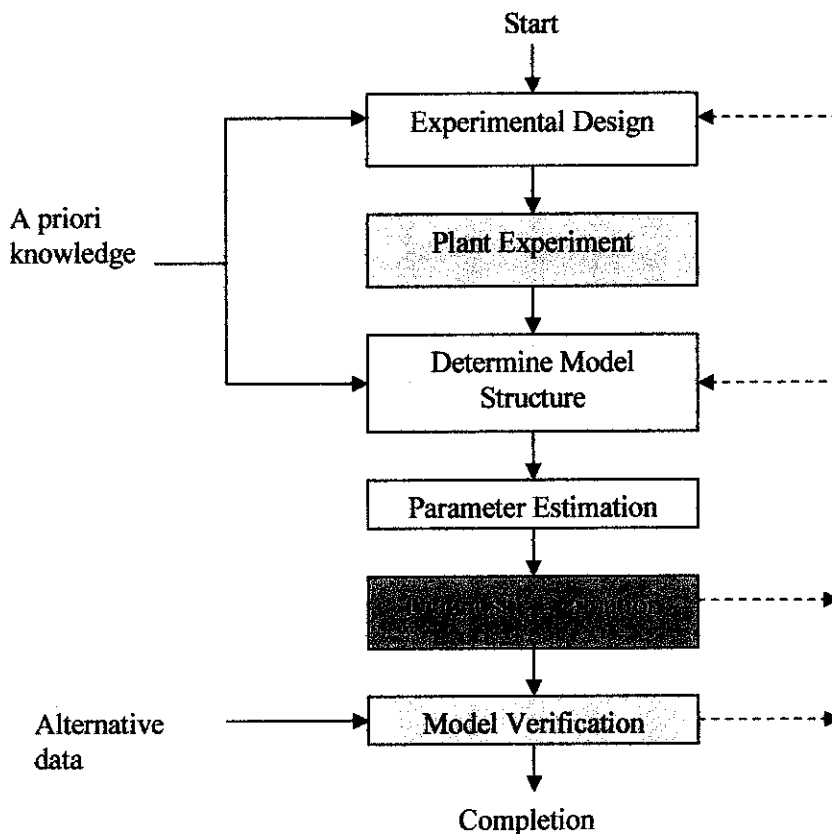


Figure 2.3: Empirical Model Building Procedure [2]

### 2.5.1 Experimental Design

As in Figure 2.3, the empirical modeling involves six-step procedure. The most important is the first procedure which is experimental design. Basically this step design will determines the shape, duration and base operating conditions for the

process. This will result in determining the condition about which the model is accurate and magnitude of the input perturbation.

### **2.5.2 Plant Experiment**

Plant experiment is executed as close as the actual plant to ensure that the disturbances during the experiment can be reduced. The operation is to be monitored continuously to verify that the output is useful for identifying a dynamic model. This is because variation in plant operation is inevitable where changes in other inputs during the experiment could make the data unusable.

### **2.5.3 Determining Model Structure**

The purpose or goal of this procedure is to develop a model that describes the input-output behavior of the process adequately for use in process control.

### **2.5.4 Parameter Estimation**

Two methodologies used to determine values for the model parameters are graphical technique and statistical principles. Both of these methods provide estimation for parameters in transfer function models, such as gain, time constant and dead time.

### **2.5.5 Diagnostic Evaluation**

This procedure is to evaluate and determine how well the model fits the data used for parameter estimation. Two approaches that can be used

- A comparison of the model prediction with the measured data
- A comparison of the result with any assumptions used in the estimation method.

### 2.5.6 Verification

This procedure is performed to be sure that typical variation in plant operation does not significantly degrade model accuracy.

### 2.6 The Process Reaction Curve

The process reaction curve method is used for identifying dynamic models. This method involves the following four actions;

- i. Allow the process to reach the steady state.
- ii. Introduce a single step change in the input variable.
- iii. Collect input and output response data until the process again reaches steady state.
- iv. Perform the graphical reaction curve calculation.

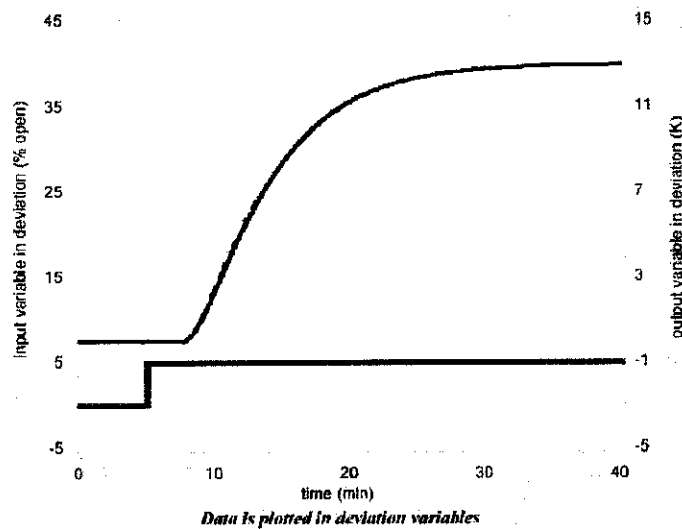


Figure 2.4: Ideal Process Reaction Curve based on the assumption of the output is first-order-with-dead-time [4]

The graphical reaction curve calculations consist of two methods namely Method I and Method II. The general overview of these methods is as follows:

### 2.6.1 Method I

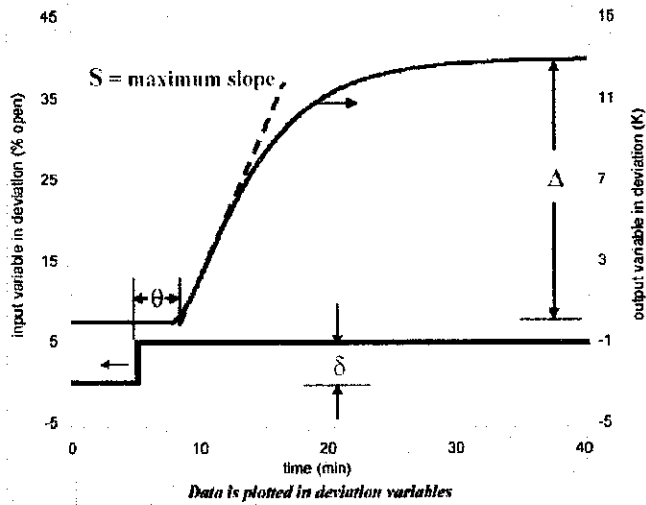


Table 2.1: Parameters to be calculated using Method I

Process gain	$K_p = \frac{\Delta}{\delta}$
Time Constant	$\tau = \frac{\Delta}{S}$
Dead time	$\theta$

Figure 2.5: Graphical approach for Method I [4]

### 2.6.2 Method II

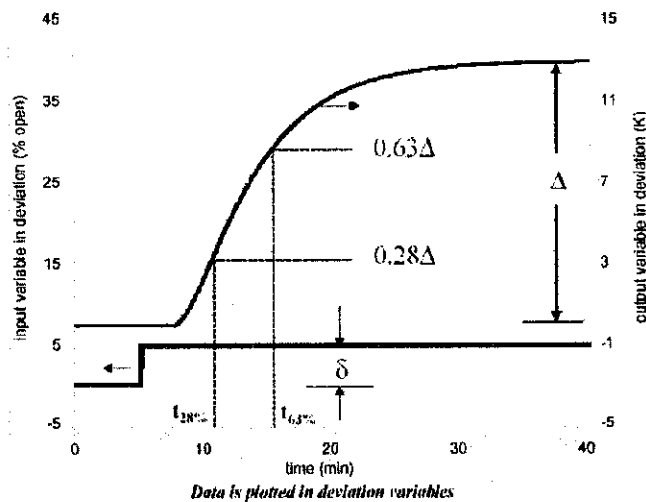


Table 2.2: Parameters to be calculated using Method II

Process gain	$K_p = \frac{\Delta}{\delta}$
Time Constant	$\tau = 1.5(t_{63\%} - t_{28\%})$
Dead time	$\theta = t_{63\%} - \tau$

Figure 2.6: Graphical approach for Method II [4]



## **2.7 Ziegler-Nichols Open-Loop Tuning**

To obtain the PID values used such as  $K_c$ ,  $T_I$  and  $T_d$ , the Ziegler-Nichols open-loop tuning based on process reaction curve is used. The open loop method is based on a measurement range of 0-100 and continuous control. This requires adjustments for other measurement ranges and for the control interval in digital systems

## **2.8 Ziegler-Nichols Closed-Loop Tuning Correlation**

Aside from using the Ziegler-Nichols open-loop tuning, Ziegler-Nichols closed-loop tuning correlation is another method that could be used to obtain the values for each  $K_c$ ,  $T_I$  and  $T_d$ . The closed loop methods does not require adjustments, a big advantage, since both process and controller are part of the test, but suffers from one major disadvantage: Bringing the loop into stable, sustained oscillation is simply out of the question for industrial processes.

## **2.9 Feedforward Controller**

The purpose of a feedforward controller is for enhancing single-loop PID control performance where disturbance is introduced. Feedforward uses the measurement of an input disturbance to the plant as additional information where this measurement provide an “early warning” that the controlled variable will be upset some time in the future.

The approach to designing a feedforward controller is based on completely canceling the effect of the disturbance. It is important to note that the feedforward controller depends on the models for the disturbance and the process. On the next page is the block diagram of the process with the addition of a feedforward controller.

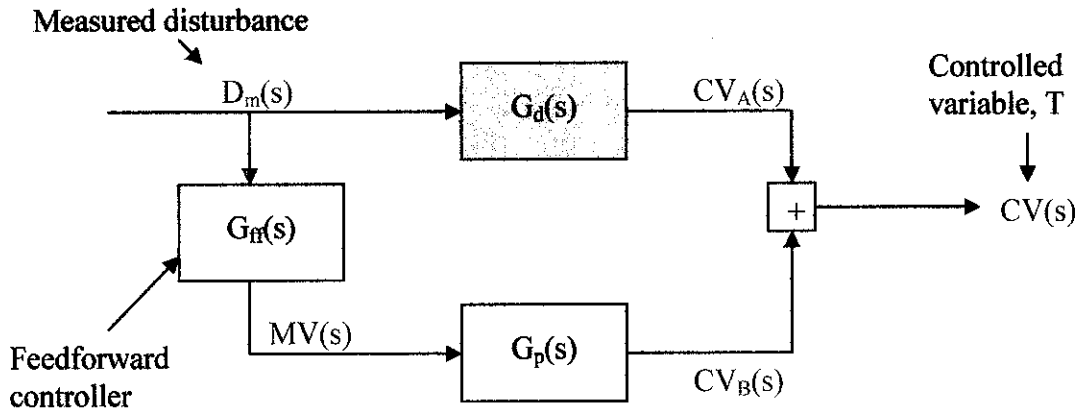


Figure 2.7: Simplified block diagram of feedforward compensation

Based on this block diagram, the equation (2.1) and equation (2.2) are obtained.

$$G_{ff}(s) = \frac{MV(s)}{D_m(s)} = -\frac{G_d(s)}{G_p(s)} \quad (2.1)$$

$$G_{ff}(s) = \frac{MV(s)}{D_m(s)} = K_{ff} \frac{T_{ld}s+1}{T_{lg}s+1} e^{-\theta_{ff}s} \quad (2.2)$$

Where:

$$\text{Lead-lag} = \frac{T_{ld}s+1}{T_{lg}s+1}$$

$$\text{Feedforward controller gain} = K_{ff} = -\frac{K_d}{K_p}$$

$$\text{Controller dead time} = \theta_{ff} = \theta_d - \theta_p \geq 0$$

$$\text{Lead time} = T_{ld} = \tau_p$$

$$\text{Lag time} = T_{lg} = \tau_d$$

## **2.10 ANFIS**

This is the major training routine for Sugeno-type fuzzy inference systems. Adaptive-  
Network-based Fuzzy Inference Systems (ANFIS) uses a hybrid learning algorithm to  
identify parameters of Sugeno-type fuzzy inference systems.

It applies a combination of the least-squares method and the back propagation  
gradient descent method for training FIS membership function parameters to emulate a  
given training data set. ANFIS can also be invoked using an optional argument for  
model validation. The type of model validation that takes place with this option is a  
checking for model overfitting, and the argument is a data set called the checking data  
set.

## **2.11 Previous Work**

The previous work is being referred to check and to understanding the steps and  
method need to be used to complete the project. So far all the steps done are similar to  
the previous project.

Previously there are two final year projects which are similar to this one. The first  
one is where the neuro-fuzzy logic controller is trained for a single SP change (0-20%  
opening of the valve) [1]. The other one is where the neuro-fuzzy logic has been trained  
with multiple set of input but it implements a switching concept [2]. Switching concept  
here means that it applies three neuro-fuzzy logic controllers with each one of them  
having data for a single SP change and operates by choosing either one of them based  
on the SP changed required.

## CHAPTER 3

### METHODOLOGY/ PROJECT WORK

#### 3.1 Procedure and Identification

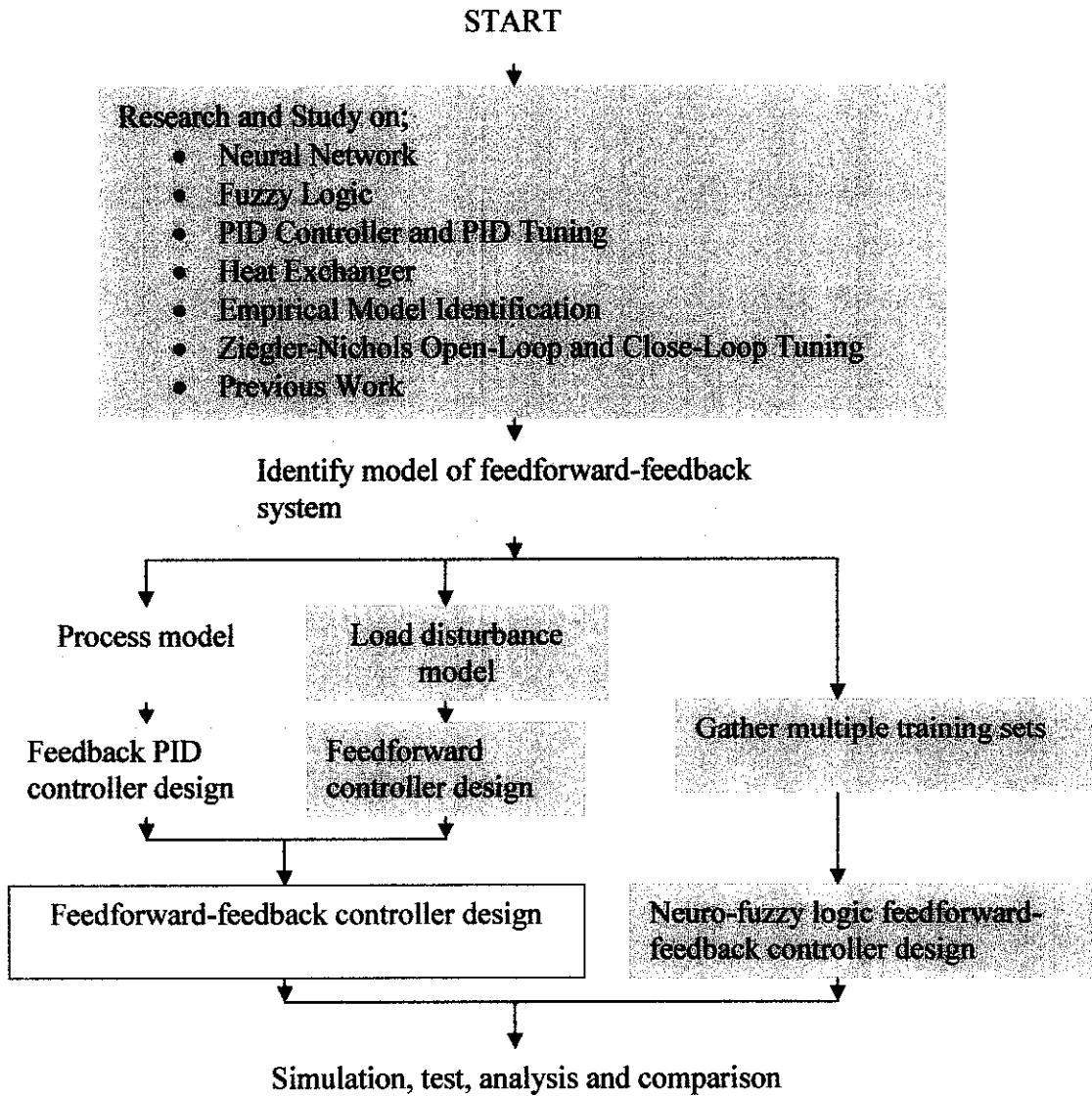


Figure 3.1: Method used to accomplish the objective of the project

### 3.2 Tools and Equipments

The PID controller is designed using Simulink in MATLAB while the neuro-fuzzy logic controller used the ANFIS Toolbox which is also in MATLAB. Aside from the software, equipment such as the SIM 305 Pilot Plant: Plant 6 (Appendix A) is used for the plant experiment and data gathering.

### 3.3 Data Gathering

The data is obtained by doing plant experiment. As briefly discuss in the introduction part, the data gathered is for the purpose of system modeling. The experiment consist of two parts; one is to obtained data for building a feedback loop and the other one is to obtained data the feedforward loop.

For the feedback loop, a small change is made at the input variable during the plant experiment and the resulting output or dynamic response is analyzed. During this experiment, the controller is set to operate in the manual mode. The experiment is done to obtained result as shown below;

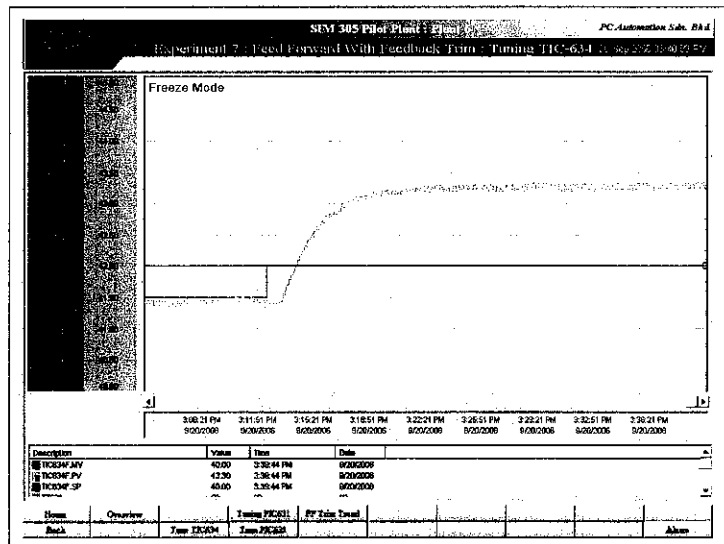


Figure 3.2: Result obtained from plant experiment

Due to some technical problem with the Distribution Control System of the Pilot Plant, the feedforward loop could not be retrieve using the plant experiment. Therefore, the data requires to design the feedforward loop (the disturbance model) is taken from the previous final year project (by Mohd Faizal Ja'afar) [1].

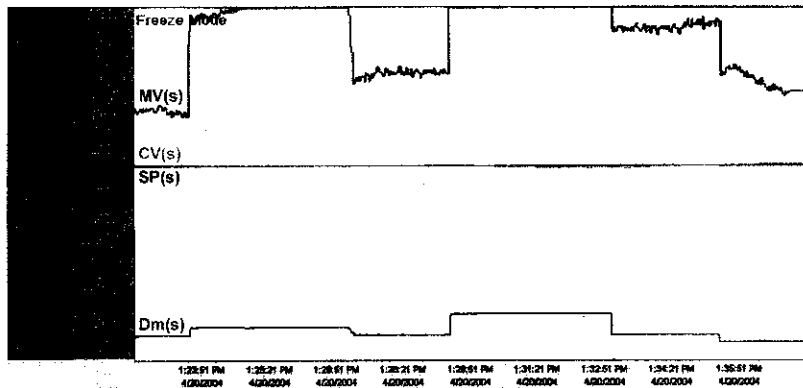


Figure 3.3: Result obtained from the previous final year project [1]

### 3.4 Data Processed

The data obtained from the experiment (the feedback loop data) is estimated using the empirical modeling two methods (refer Chapter 2). From these methods, two transfer functions will be obtained.

For the other data (the one retrieved from the previous final year project) [1], it is used to generate a disturbance model and a feedforward controller.

### 3.5 PID Tuning

To obtain the PID values used such as  $K_c$ ,  $T_I$  and  $T_d$ , the Ziegler-Nichols open-loop tuning and Ziegler-Nichols closed-loop tuning correlation based on process reaction curve is used;

Table 3.1: Ziegler-Nichols open-loop tuning

	$K_c$	$T_I$	$T_d$
P-only	$(1 / K_p) / (\tau / \theta)$	-	-
PI	$(0.9 / K_p) (\tau / \theta)$	$3.3\theta$	-
PID	$(1.2 / K_p) (\tau / \theta)$	$2.0\theta$	$0.5\theta$

Table 3.2: Ziegler-Nichols closed-loop tuning correlation

	$K_c$	$T_I$	$T_d$
P-only	$K_u / 2$	-	-
PI	$K_u / 2.2$	$P_u / 1.2$	-
PID	$K_u / 1.7$	$P_u / 2$	$P_u / 8$

Where  $K_u$  = ultimate gain, and  
 $P_u$  = ultimate period

The ultimate gain and the ultimate period are obtained by running the simulation in P-mode. The proportional value is varied until CV reaches a sustained oscillation while the others, the integral and the derivative remain zero. Sustained oscillation means that the oscillation did not grow nor decay but maintains at constant amplitude. The proportional value that enables CV to reach the sustained oscillation is the ultimate gain while the ultimate period is the time required for the sustained oscillation to complete one oscillation cycle.

### 3.6 Disturbance Model and Feedforward Controller

The load disturbance has to be identified before the disturbance model and the feedforward controller can be implemented. In order to do so, all measured process variable that are capable of being the load disturbance are evaluated with 5

Feedforward Variable Selection Criteria. Only the variable that satisfies all selection criteria will be considered as the load disturbance variable, in which its model will then be empirically obtained.

The feedforward controller is build based on the disturbance model construct by the previous final year project (by Muhammad Faizal Ja'afar). In order to design the disturbance model, the disturbance variable has to be identified first. Identification is done by evaluating all measured potential variables in the pilot plant using the Feedforward Variable Selection Criteria [1].

Table 3.3: Evaluation of Potential Feedforward Variables [1]

Criteria	Potential Disturbance Variables *				
	TT-631	FT-631	TT-632	TT-633	FT-664
Single loop control <i>not satisfactory</i>	YES	YES	YES	YES	YES
The variable is <i>measured</i>	YES	YES	YES	YES	YES
The variable indicates the <i>key disturbance</i>	NO	NO	NO	NO	YES
<i>No causal relationship</i> between MV and the feedforward variable	NO	NO	NO	YES	YES
Variable dynamics is <i>not significantly faster</i> than MV dynamics	NO	NO	NO	YES	YES



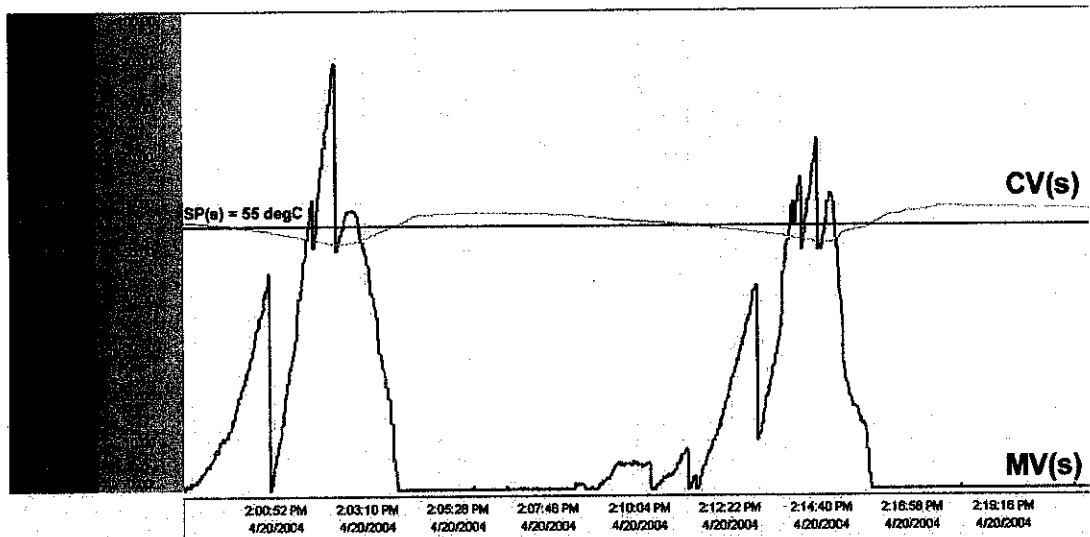


Figure 3.4: Feedback-Only Control Performance [1]

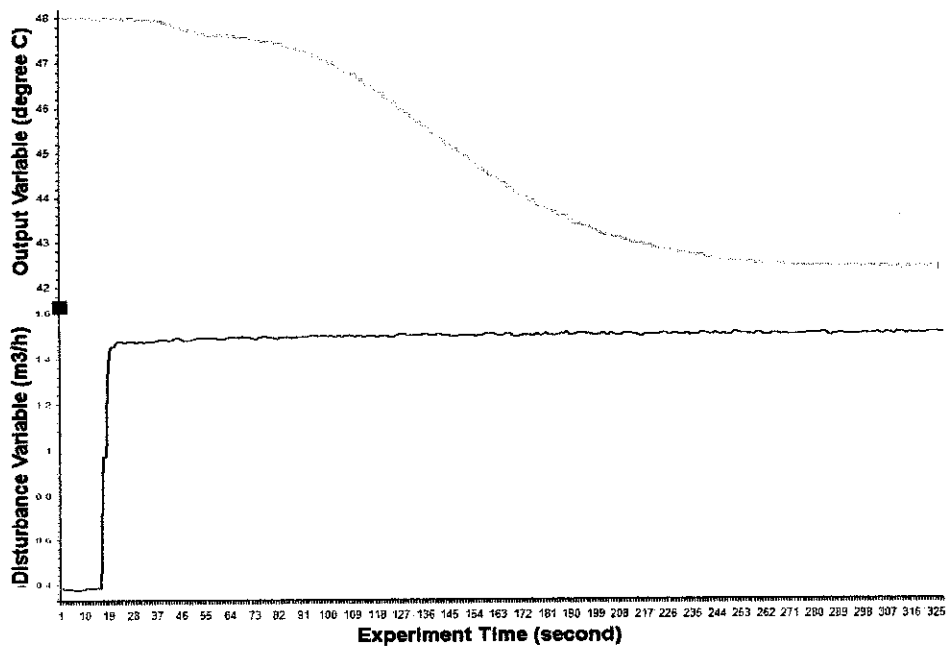


Figure 3.5: Load Disturbance Reaction Curve Plotted from Data [1]

The load disturbance model parameters are calculated as follows: -

- Disturbance gain,  $K_d$ : -

$$= \frac{\Delta}{\delta} = \frac{(42.4^{\circ}\text{C} - 47.9^{\circ}\text{C})}{(4.17 \times 10^{-4} \text{ m}^3/\text{s} - 1.11 \times 10^{-4} \text{ m}^3/\text{s})} = \frac{-5.5^{\circ}\text{C}}{3.06 \times 10^{-4} \text{ m}^3/\text{s}} = \underline{\underline{-17973.856}} \quad [1]$$

- Disturbance Dead time,  $\theta$  : -  
= from observation = **40 seconds** [1]

- Disturbance Time Constant,  $\tau_d$ : -  
=  $1.5(t_{63\%} - t_{28\%}) = 1.5(156 - 113 \text{ seconds}) = 1.5(43) = \mathbf{64.5 \text{ seconds}}$  [1]

### 3.7 Simulation Block and Testing

Two controller designs are being used to test the parameters obtained using Ziegler-Nichols open-loop tuning and Ziegler-Nichols closed-loop tuning correlation. The designs are as in Figure 3.6 and Figure 3.7 below. The 'Out1' display the step change, the controlled variable (CV) and the process variable (PV).

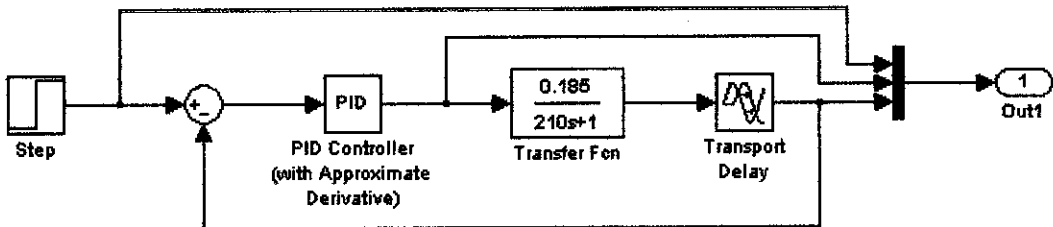


Figure 3.6: The control process block diagram using a single block PID controller

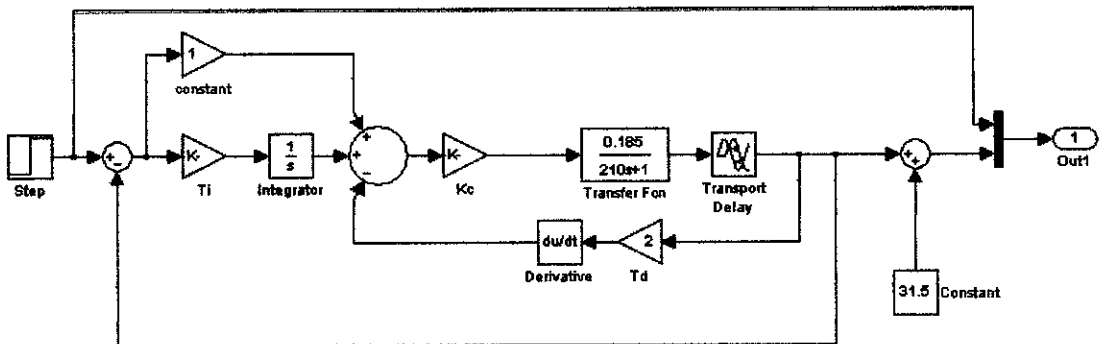


Figure 3.7: Similar control process block diagram but using different PID controller

The major difference between these two designs is in the PID algorithm where the equation (3.7.1) represents the PID block of Figure 3.6 and equation (3.7.2) represents the PID block of Figure 3.7. The PID presented in Figure 3.7 is chosen since the equation is recommended and proves to be more sufficient than in Figure 3.6.

$$MV(s) = P + \frac{I}{s} + Ds \quad (3.7.1)$$

$$MV(t) = K_c \left( E(t) + \frac{1}{T_i} \int_0^t E(t') dt' - T_d \frac{dCV(t)}{dt} \right) + I \quad (3.7.2)$$

The controller is evaluated based on its performance. The performance can be measured using several indicators. Figure 3.8 shows an example of SP change and the resultant CV. From this figure, two indicators can be used to measure the controller performance that is: (i) the decay ratio and (ii) CV overshoot.

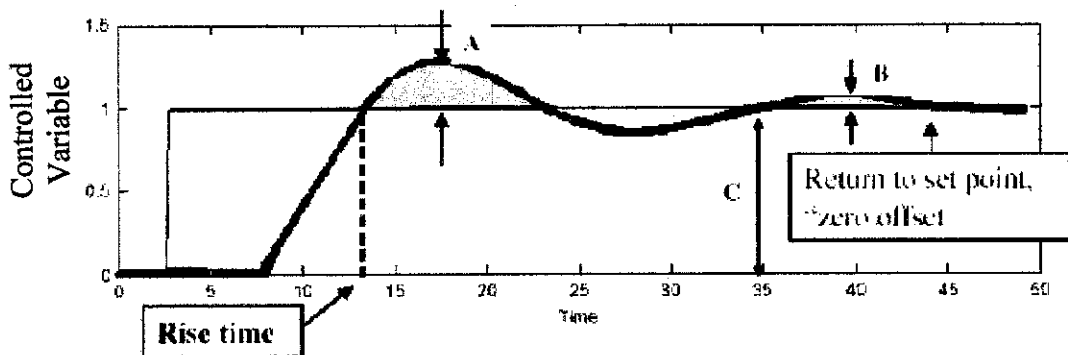


Figure 3.8: Typical CV response of a feedback control system to a step SP change

### 3.7.1 The decay ratio

The decay ratio (DR) is the ratio of neighboring peaks in an underdamped CV response. Usually, periodic behavior with large amplitudes is avoided in process variables; therefore, a small DR is usually desired, and an overdamped response is sometimes desired. Based on Figure 3.8, the DR can be calculated:

$$DR(\%) = \frac{B}{A} \times 100\%$$

where A = 1<sup>st</sup> peak

B = 2<sup>nd</sup> peak

The performance of the controller can only be accepted or classified as good if the DR is less than 25%.

### 3.7.2 CV overshoot

CV OS measures the performance at the output of the process. This quantity is important as it determines the overall process performance. With large variations, the process could take longer time to settle or reach steady-state and might not be appropriate for certain process. Therefore a small CV OS is more appropriate and applicable in process application. Based on Figure 3.8, the CV OS can be calculated:

$$CV(\%) = \frac{A}{C} \times 100\%$$

where A = the highest value CV can exceed from the final steady-state value

C = the final steady-state value

The performance of the controller is acceptable if the CV OS is less than 20%.

### 3.7.3 MV overshoot

Different with DR and CV OS, MV OS measures the performance at the output of the controller. This quantity is of concern because the MV is also a process variable that influences performance. There are often reasons to prevent large variations in the MV. Some large variations can cause long-term degradation in equipment performance. In other cases manipulations can disturb an integrated process. The overshoot of MV is used to indicate how aggressive the controller has been adjusted. The overshoot is the maximum amount that the MV exceeds its final steady-state value and is usually expressed as a percent of the change in MV from its initial to its final value.

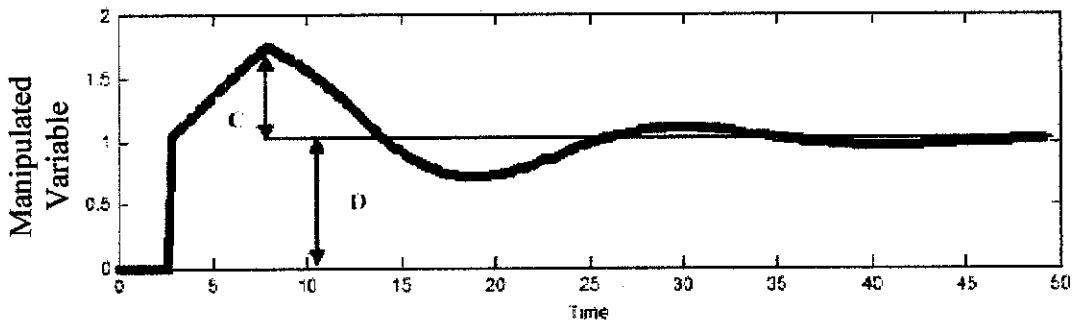


Figure 3.9: Typical MV response of a feedback control system to a step SP change

From the Figure 3.9, the MV OS can be calculated:

$$MV(\%) = \frac{C}{D} \times 100\%$$

where C = the highest value MV can exceed from the final steady-state value

D = the final steady-state value

The performance of the controller can only be accepted or classified as good if the MV OS is between 50% - 150%.

### 3.8 Generating Neuro-Fuzzy Logic Controller

To build the fuzzy-logic controller, a few steps need to be taken, especially regarding the input to the fuzzy-logic block. An additional block is inserted into the simulation in order to import the data required to the fuzzy-logic toolbox (ANFIS). In Figure 3.10, a block is added that enables the data to be transferred to the workspace before it could be called and used as training data of the ANFIS.

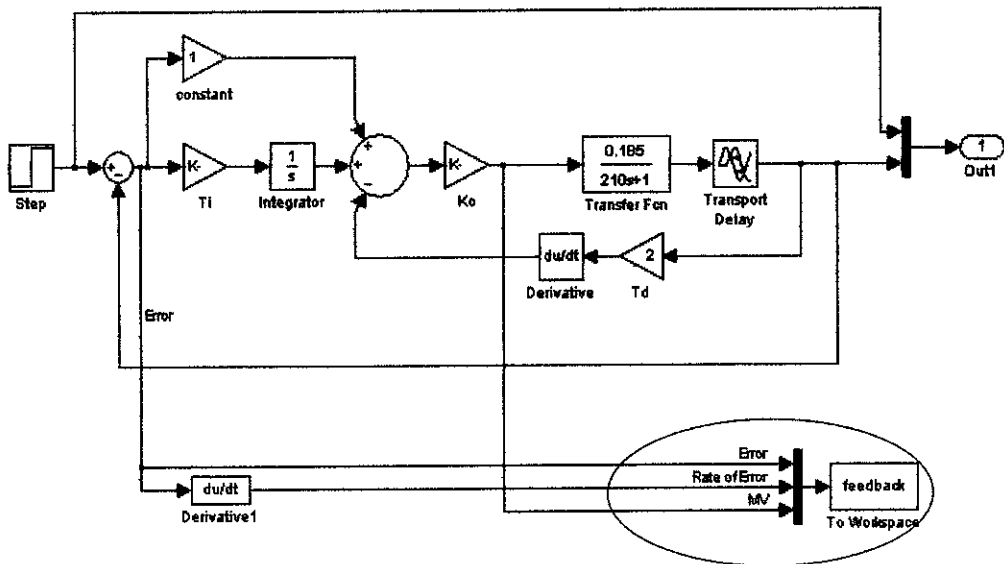


Figure 3.10: An additional block added for the purpose of importing data to the workspace

In Figure 3.11 below is the simplified methodology of generating an .fis file which is used by the fuzzy-logic block in Simulink.

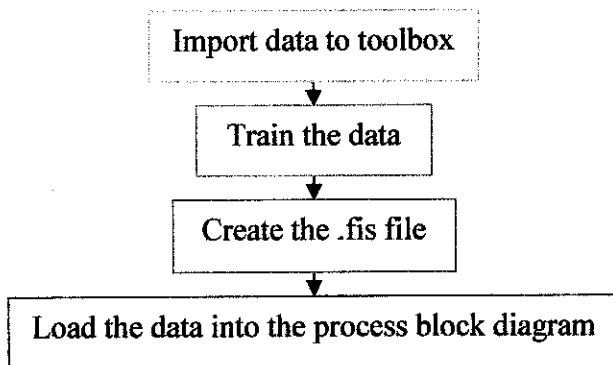


Figure 3.11: Simplified methodology to implement the neuro-fuzzy logic controller

### 3.8.1 Data Import to ANFIS

The first step of creating the controller is by loading the data onto the ANFIS toolbox. The data is presented in graphical form in Figure 3.12 below.

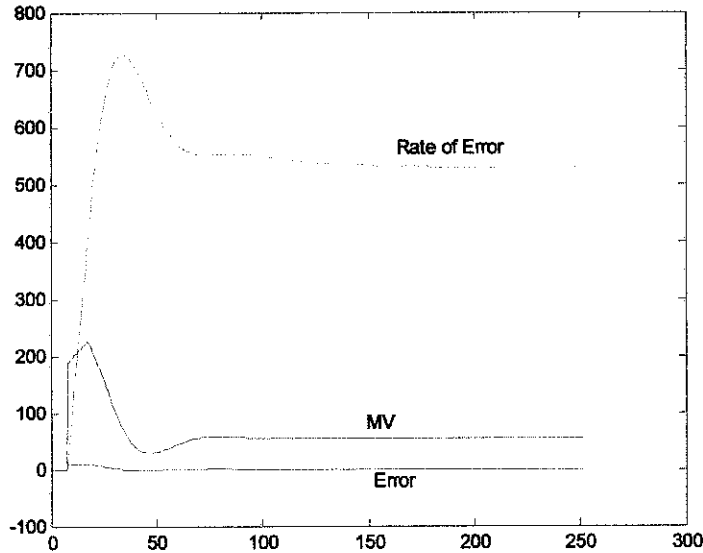


Figure 3.12: Data to be used for training the neuro-fuzzy logic controller

When these data is loaded into the ANFIS toolbox, the plotted data is shown in Figure 3.13. Basically the data plotted is the output response. In this case, MV is the output response while the error and rate of error is the input.

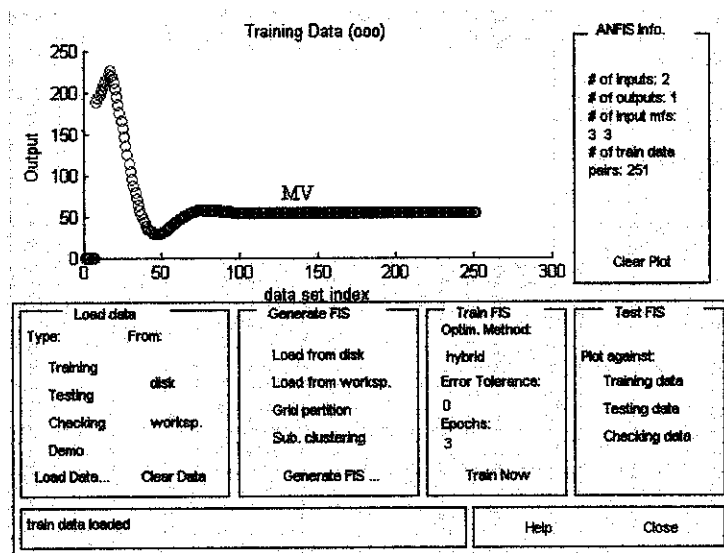


Figure 3.13: The plotted MV as it is loaded into the ANFIS toolbox

### 3.8.2 Train the Neuro-Fuzzy Logic Controller

Before the data is trained, a networking between neurons needs to be established between the input data (the error and rate of error). By generating the FIS structure in the ANFIS toolbox, a mapping structure is established as in Figure 3.14.

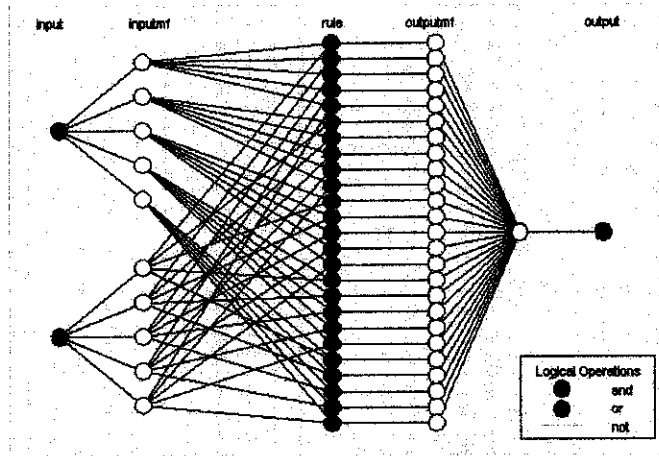


Figure 3.14: The mapping structure obtained by generating the FIS.

Since ANFIS toolbox is used, the relationship between the inputs and the output are defined automatically by the toolbox. No changes need to be done to the rules or relationship. The next step is start training the data and obtains the result for the neuro-fuzzy logic controller.

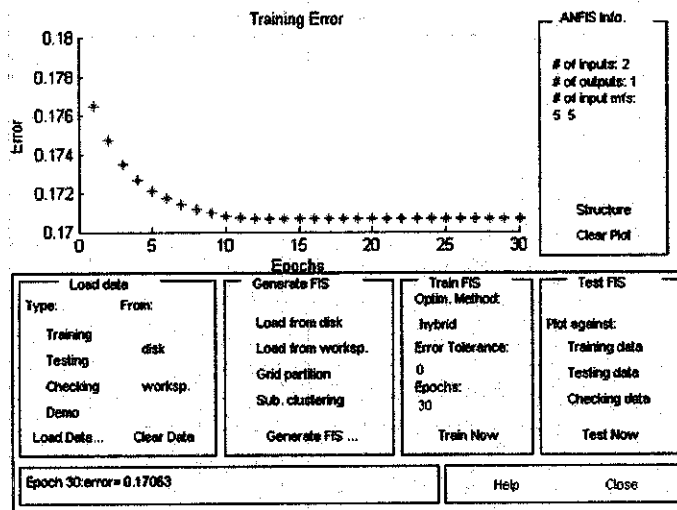


Figure 3.15: The plotted training result reduces error close to zero



### 3.8.3 Creating the .fis File

After the training is completed, the trained neuro-fuzzy logic controller needs to be saved as a .fis file. This is done to enable the neuro-fuzzy logic controller to be used in the simulation.

### 3.8.4 Load the Trained Neuro-Fuzzy Logic Controller

In order to run the simulation using this controller, it has to be exported to MATLAB. To do so, a MATLAB command needs to be executed as describe below;

```
>>fuzzy {file name.fis}
```

By doing this, a window will appear as in Figure 3.16.

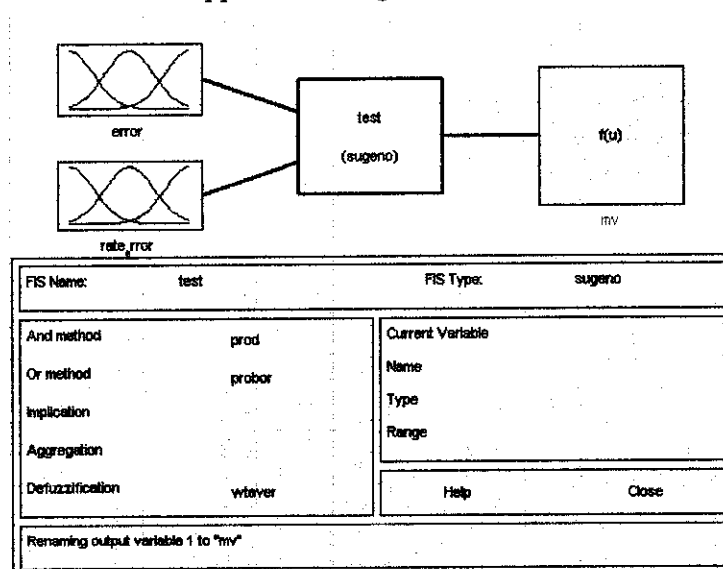


Figure 3.16: The neuro-fuzzy logic relationship between the input and output based on Sugeno type

The data can now be exported to the workspace. This can be done by:

```
→File → Export → To workspace
```

As in Figure 3.17, the Simulink model can now be simulated when the fuzzy logic controller has been loaded with the imported data.

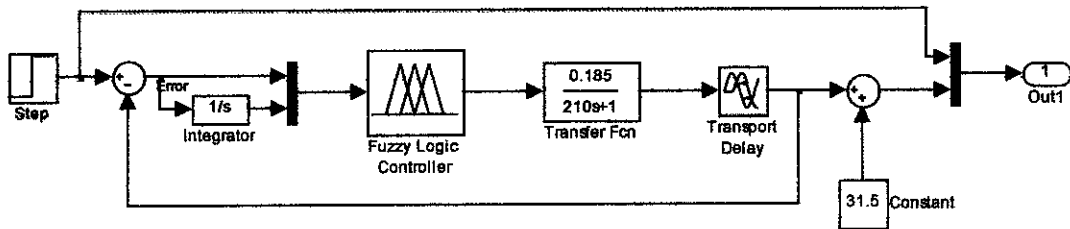


Figure 3.17: A new process block diagram where PID is replaced with fuzzy-logic controller

### 3.9 Design a training approach for the fuzzy logic controller to accept training data more than one

From the two previous final year projects reviewed, in [1], the author managed to implement the fuzzy logic controller for a single set point while the other, [2] uses a switching concept to change the input/ output of the fuzzy logic controller for a few ranges of set point. This project aims to implement the fuzzy logic controller for a certain range of set point without the use of switching.

Predetermined variable:-

Initial temperature	= 31.5°C
Maximum heat exchanger operating temperature	= 70°C
Range of SP change	= 0°C - 40°C

The maximum SP change is set to 40°C since the maximum value achievable for the heat exchanger in the pilot plant is 70°C. For these training purpose two alternatives has been tried out (Alternative 1 and Alternative 2).

### 3.9.1 Alternative 1

The first method tried is by implementing a method as illustrated in the Figure 3.18 below.

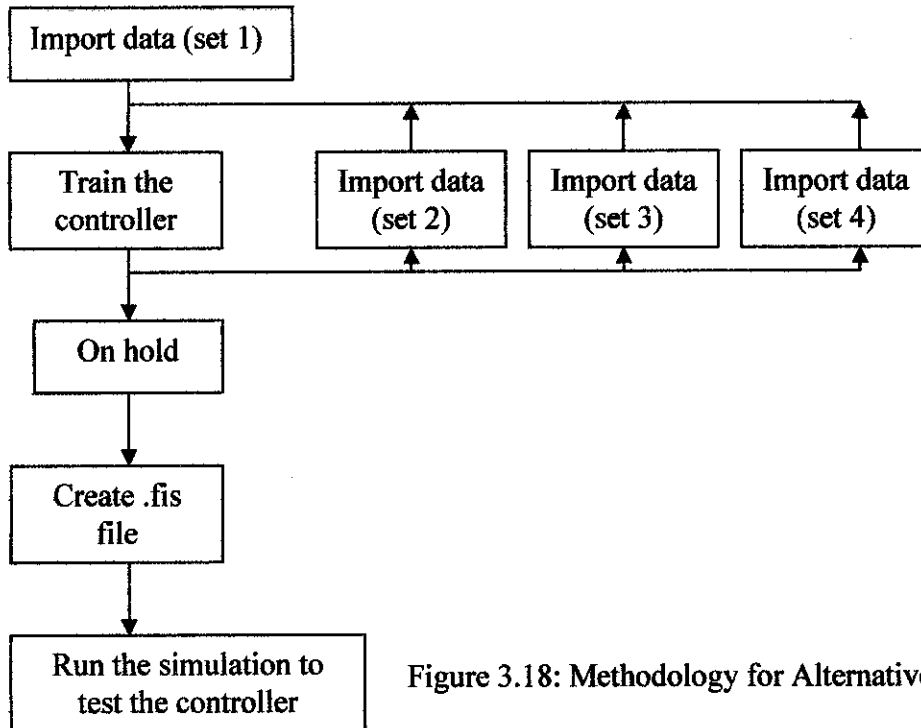


Figure 3.18: Methodology for Alternative 1

In the method above, 4 sets of training data are collected where each data represents a different change of set point:

- Set 1(0°C - 10°C)
- Set 2(0°C - 20°C)
- Set 3(0°C - 30°C)
- Set 4(0°C - 40°C).

### 3.9.2 Alternative 2

The second method tried is by implementing a method as illustrated in the Figure 3.19 below.

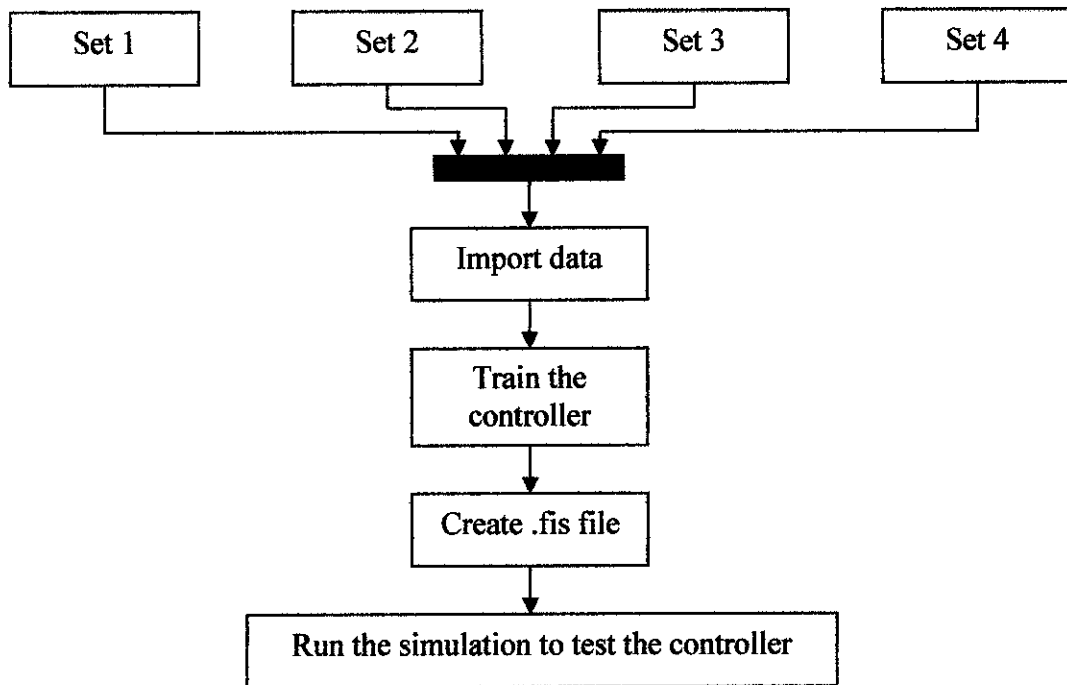


Figure 3.19: Methodology for Alternative 2

Using the same set of data, another method is tried out where all the data is first gathered before it is imported to the ANFIS toolbox. This method requires a little bit of work where the data is arranged in one array and then exported to the ANFIS toolbox. Below is the detailed description of the method since after some simulation test, the resulting output does correspond with the desired output.

## **CHAPTER 4**

### **RESULT AND DISCUSSION**

#### **4.1 The Process Transfer Function**

Using the data from plant experiments, two transfer functions are obtained based on Method I and Method II (refer Chapter 2).

##### **Transfer function 1 (TF1)**

$$\frac{Y(s)}{X(s)} = \frac{0.185e^{-42s}}{210s+1}$$

##### **Transfer function 2 (TF2)**

$$\frac{Y(s)}{X(s)} = \frac{0.185e^{-84s}}{90s+1}$$

Each of the transfer function is tested in open loop approach in order to determine which one closely represents the process reaction curve (plant experiment). The percentage or the rate of difference between the models is used to compare between these two transfer functions. The one yields the less diff is chosen. TF1 is chosen as the transfer function because it yields 51.16% difference compared to TF2 with 51.46%. The values are rather high because the calculation is done by estimating the data since the actual data is corrupted (refer to Appendix B).

#### **4.2 Testing the PID Controller**

As mentioned in Chapter 2, two tuning methods have been used in this project to determine the PID controller parameters. The calculated PID parameters are as in the

Table 4.1 and Table 4.3 below. The simulation has been run in three modes (P-only, PI and PID) to verify which mode satisfy and yield the most effective result.

Table 4.1: Calculated variables for PID using Ziegler-Nichols Open-loop tuning

	$K_C$	$T_I$	$T_d$
P-only	1.08108	-	-
PI	24.32432	138.6	-
PID	32.43243	84	21

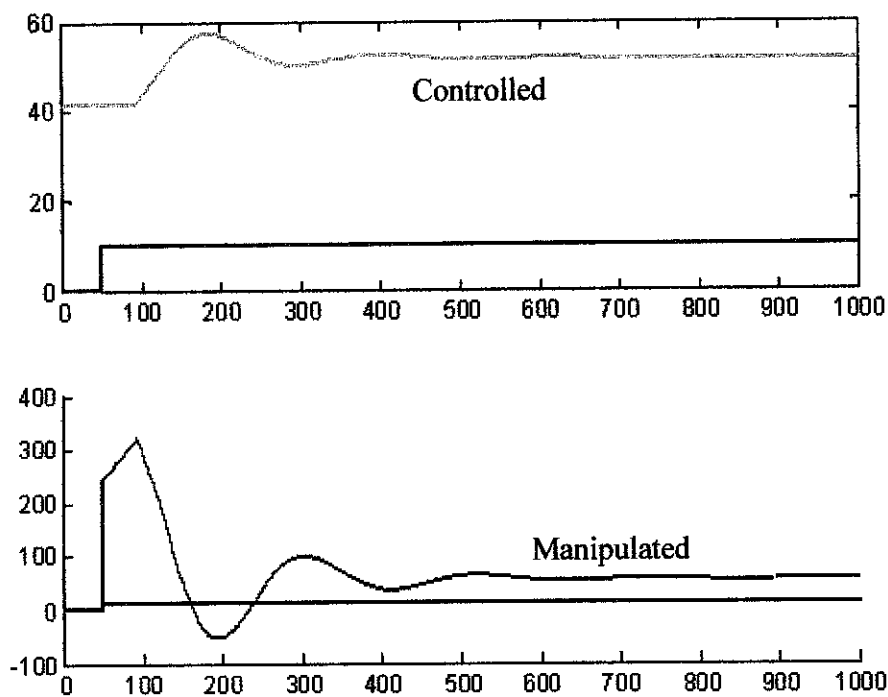


Figure 4.1: The transient response of the feedback control system to a step set point change (using Z-N Open-Loop Tuning)

Figure 4.1 shows the resulting simulation using Ziegler-Nichols Open-loop tuning method. The measured performance is as in Table 6 below:

Table 4.2: The performance check using Z-N Open-Loop Tuning

Indicator	Values
DR	= 17%
CV OS	= 54.75%
MV OS	= 593.63%

Table 4.3: Calculated variables for PID using Ziegler-Nichols Closed-Loop tuning correlation

	$K_C$	$T_I$	$T_d$
P-only	23	-	-
PI	20.909	129.4248	-
PID	27.0588	77.6549	19.4137

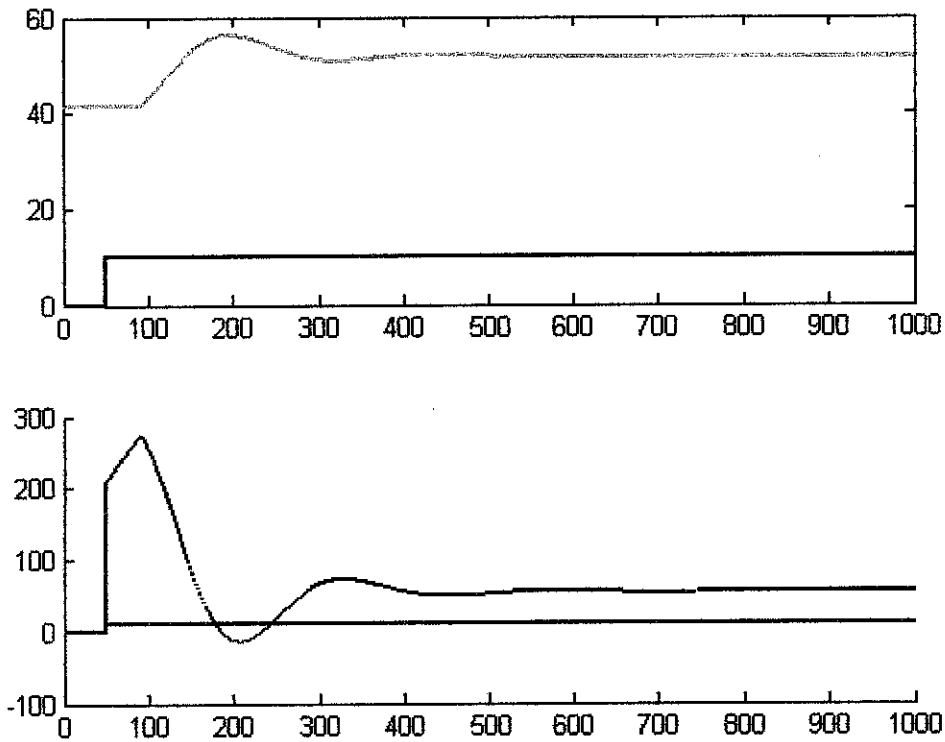


Figure 4.2: The transient response of the feedback control system to a step set point change (using Z-N Closed-Loop Tuning Correlation)

Figure 4.2 shows the resulting simulation using Ziegler-Nichols Closed-loop tuning correlation method. The measured performance is as in Table 8 below:

Table 4.4: The performance check using Z-N Closed-Loop Tuning Correlation

Indicator	Values
DR	= 8.69%
CV OS	= 44.9%
MV OS	= 506.04%

From the result obtained, the PID parameters obtained using Ziegler-Nichols Closed-Loop Tuning Correlation is used since it yields less percentage in every performance indicator measured. These parameters requires further fine tuning in order to have the desired performance within the setting range (DR less than 25% and CV OS less than 20%).

### 4.3 Fine Tuning

Fine tuning is conducted to modify the PID parameters so that it could perform to the required output. The objective is to adjust the parameters so that the CV obtained have less than 20% overshoot and the decay ratio is less than 25%.

Table 4.5: The initial values to be used for fine tuning

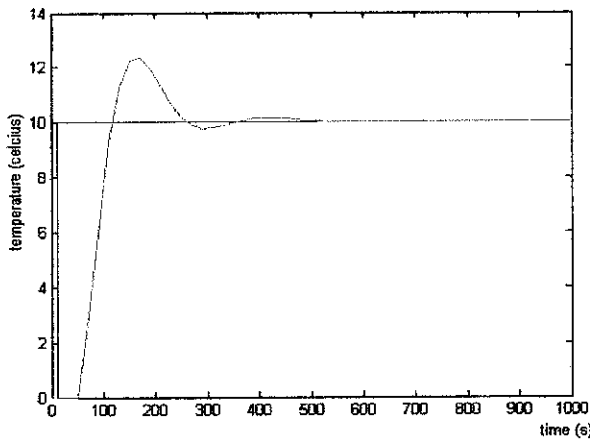
	$K_C$	$T_I$	$T_d$
PI	20.909	129.4248	-

The fine tuning emphasizes the concept of trial and error. The tuning process requires one parameter to be constant while the other one is tuned. This is done to observe the effect of changing one parameter to the output of the process. Below is the result of tuning the PID controller and CV response.

Table 4.6: The values obtained after the fine tune

	Initial	After Tuning
P ( $K_C$ )	20.909	18.5
I ( $T_I$ )	129.4248	180





CV overshoot = 23.4%

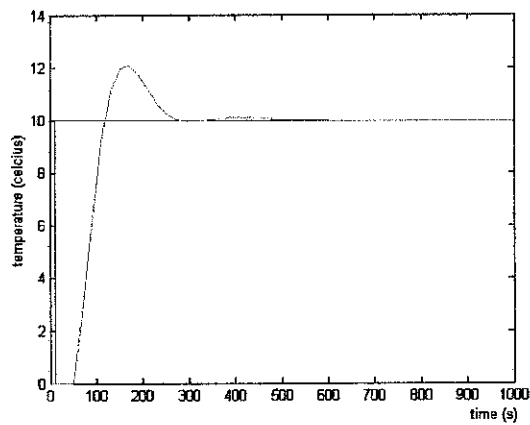
Decay ratio = 6%

Figure 4.3: Tuning using PI-mode

It is decided that the derivative mode should be activated and the impact is it reduces the CV overshoot peak.

Table 4.7: The finalized values obtained after the fine tune with addition of  $T_D$

	Initial	After Tuning
P ( $K_C$ )	20.909	18.5
I ( $T_I$ )	129.4248	180
D ( $T_D$ )	0	2



CV overshoot = 21.0%

Decay ratio = 13.5%

Figure 4.4: Tuning using PID-mode

The result obtained after tuning the PID controller above is closer to the conditions set priority i.e. (CV OS  $\leq$  20% and DR  $\leq$  25%) however based on the graph shown in Figure 4.4, the settling time is longer compared to the PI-mode.

#### 4.4 Feedforward Controller

The feedforward controller is build based on the disturbance model construct by the previous final year project (by Muhammad Faizal Ja'afar).

Feedforward controller is not a PID algorithm, but is a control equation that relates the load disturbance and the process. Calculation on the feedforward control equation is shown below: -

- |                                         |  |                                         |
|-----------------------------------------|--|-----------------------------------------|
| ▪ Feedforward controller gain, $K_{ff}$ |  | $= -K_d / K_p$                          |
|                                         |  | $= -(-17973.856 / 0.185)$               |
|                                         |  | $= \underline{\underline{97155.97838}}$ |
  
- |                                                   |  |                                                                         |
|---------------------------------------------------|--|-------------------------------------------------------------------------|
| ▪ Feedforward controller dead time, $\theta_{ff}$ |  | $= \theta_d - \theta_p$                                                 |
|                                                   |  | $= 40 - 42$                                                             |
|                                                   |  | $= \underline{\underline{0 \text{ second}}}$ since $\theta_{ff} \geq 0$ |
  
- |                                                |  |                                                 |
|------------------------------------------------|--|-------------------------------------------------|
| ▪ Feedforward controller lead time, $T_{lead}$ |  | $= \tau$                                        |
|                                                |  | $= \underline{\underline{210 \text{ seconds}}}$ |
  
- |                                              |  |                                                  |
|----------------------------------------------|--|--------------------------------------------------|
| ▪ Feedforward controller lag time, $T_{lag}$ |  | $= \tau_d$                                       |
|                                              |  | $= \underline{\underline{64.5 \text{ seconds}}}$ |
  
- |                      |  |                                                      |
|----------------------|--|------------------------------------------------------|
| ▪ Lead/lag algorithm |  | $= (T_{lead}s + 1) / (T_{lag}s + 1)$                 |
|                      |  | $= \underline{\underline{(210s + 1) / (64.5s + 1)}}$ |

Thus, the feedforward control equation is as follows: -

$$G_{ff}(s) = -\frac{G_d(s)}{G_p(s)} = K_{ff} \left( \frac{T_{lead}s + 1}{T_{lag}s + 1} \right) e^{-\theta_{ff}s}$$

$$G_{ff}(s) = 97155.97838 \left( \frac{210s + 1}{64.5s + 1} \right)$$

The block diagram in Figure 4.5 is the block diagram with the addition of disturbance model and feedforward controller. With the addition of these block diagram, the process loop is completed.

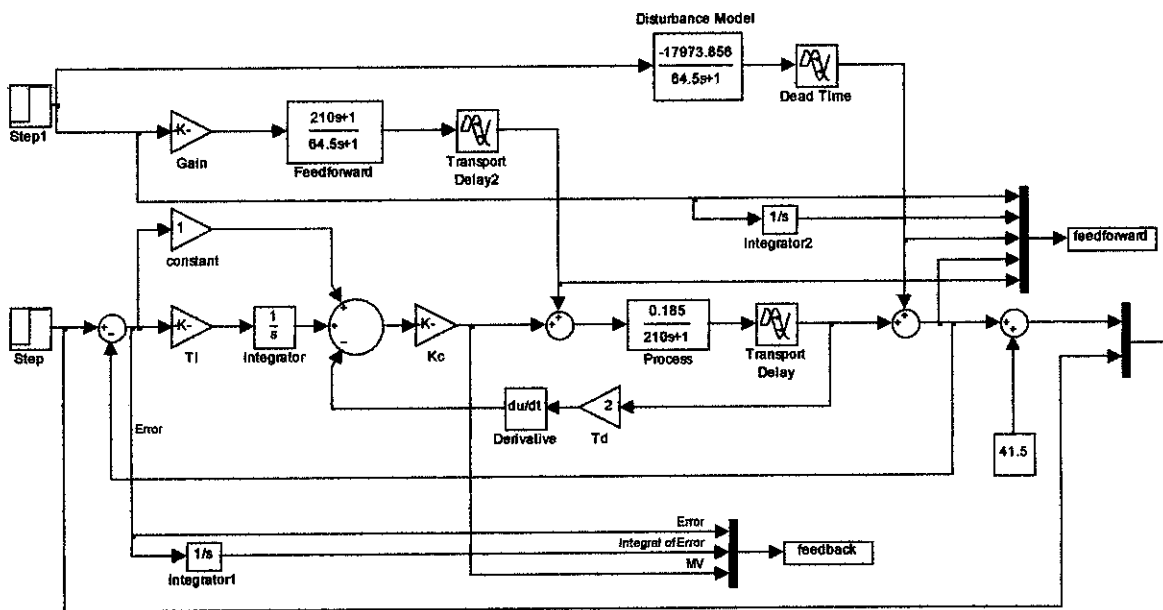


Figure 4.5: The complete process block diagram

Two tests are conducted to observe the performance of the feedforward controller against the disturbance. The first test is done by disconnecting the feedforward controller from the process to observe the effect of the disturbance (Figure 4.6). The second test is conducted with the feedforward controller connected to the process to observe how effective the feedforward controller overcomes the disturbance (Figure 4.7).

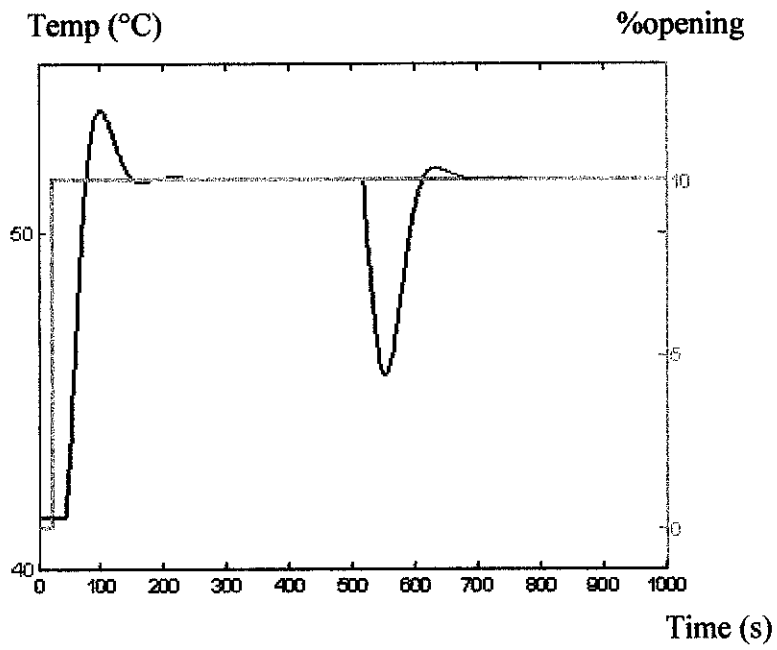


Figure 4.6: The effect of disturbance to the process

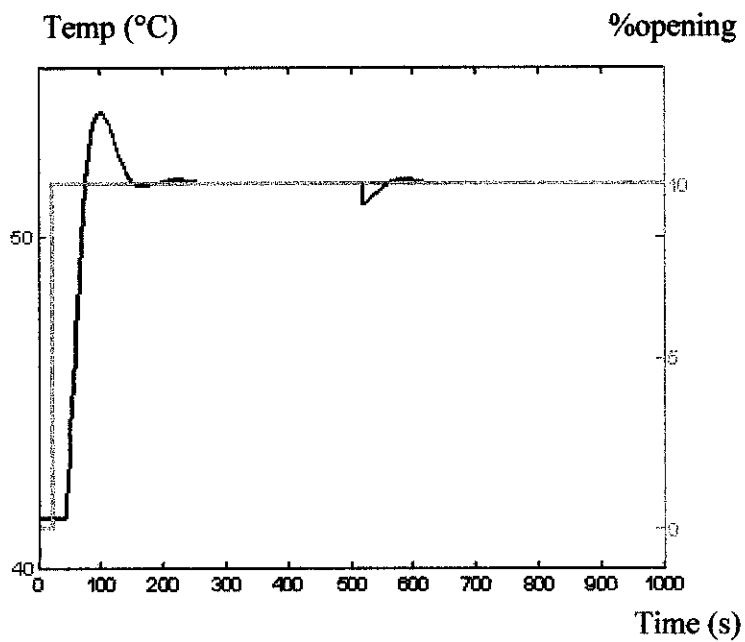


Figure 4.7: The effect of disturbance to the process is reduced by the feedforward controller

## **4.5 Training Feedback Neuro-Fuzzy Logic Controller**

The neuro-fuzzy logic controller is trained using 4 sets of training data:

- Set 1 ( 0% - 10% )
- Set 2 ( 0% - 20% )
- Set 3 ( 0% - 30% )
- Set 4 ( 0% - 40% )

### **4.5.1 Alternative 1 vs Alternative 2**

Two alternatives have been proposed earlier in Chapter 3. Each alternative has been tried out a few times and being compared to choose which is the most suitable for the training purpose. From trials, it seems that Alternative 1 is not suitable to be used. This is because by doing multiple training to the ANFIS toolbox, the recent data inserted and train will be the one used as the default data and all the previous data inserted will be deleted automatically. This means that the fuzzy logic controller will only corresponds to set 4, where the output will reach steady-state at about 71.5°C. Using Alternative 1, the controller can only be used for a single SP change related to the latest training set. Other SP changes will cause the CV to be oscillatory and does not reach zero offset steady-state.

### **4.5.2 Alternative 2 to Train the Controller**

Alternative 2 is the method where all data are arranged and combined to form a compiled set of data. These data is the combination of data from Set 1 to Set 4. In Figure 4.8, the compiled training data is shown loaded onto ANFIS toolbox.

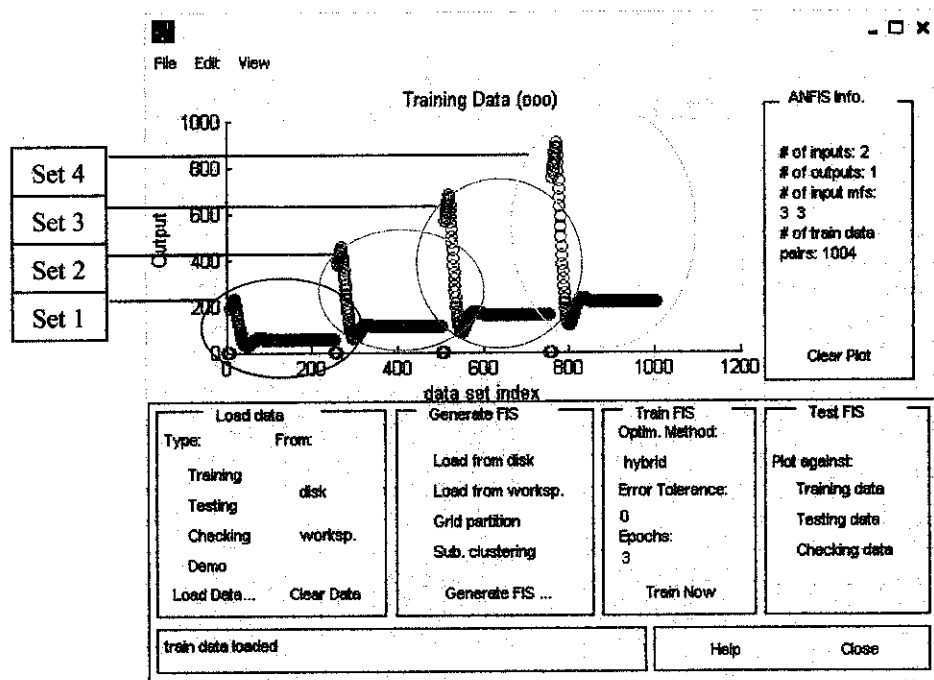


Figure 4.8: How the training set is input into the ANFIS toolbox

This method overcomes the weakness of the first method (Alternative 1) where each data inserted into the ANFIS toolbox are stored and none of them is overwritten.

#### 4.6 Performance Check and Comparison

The neuro-fuzzy controller is simulate with multiple step change to observe the performance and it is being compared with the PID controller. Two variables are used as benchmarks (CV overshoot and decay ratio) to compare or evaluate the controllers.

Table 4.8: Performance check with step change of 23% of valve opening

Step change = 0 % - 23 %		
<b>PID Controller</b>		<b>CV overshoot:</b> <u>8.9%</u>  <b>Decay ratio:</b> <u>6.19%</u>
<b>Fuzzy-Logic Controller</b>		<b>CV overshoot:</b> <u>9.14%</u>  <b>Decay ratio:</b> <u>6.43%</u>

Table 4.9: Performance check with step change of 15% of valve opening

Step change = 0 % - 15 %		
<p style="text-align: center;"><b>PID Controller</b></p>		<p>CV overshoot: <u>6.8%</u></p> <p>Decay ratio: <u>6.33%</u></p>
<p style="text-align: center;"><b>Fuzzy-Logic Controller</b></p>		<p>CV overshoot: <u>7.05%</u></p> <p>Decay ratio: <u>5.8%</u></p>

Based on Table 4.8 and Table 4.9; the performance of the neuro-fuzzy logic controller is compared with the PID controller. From these result, it shows that the performance of the controller is comparable to the PID controller since the CV overshoot and the decay ratio are nearly the same.

Based on the overshoot it can be concluded at this stage that the PID controller is better since the overshoot is less compared to the neuro-fuzzy logic controller.



However, the neuro-fuzzy logic controller can now perform at difference SP change. The difference in performance maybe due to the range of training sets. The neuro-fuzzy logic controller might work at the same level or better that the PID controller if the number of training set increases.

#### 4.7 Neuro-Fuzzy Logic Controller for Feedforward Controller

The neuro-fuzzy logic controller for the feedforward system does not yield a good result as for feedback controller. This probably because the feedforward controller is not a PID controller likes the feedback controller.

The ANFIS toolbox failed to build a network between the data that relates to the feedforward. This might be due to the nature of the feedforward controller where the corrective action is done before the disturbance occurs. The ANFIS could not recognize a sudden change of one data when the others are constant.

Table 4.10: Data used to train neuro-fuzzy logic controller for feedforward controller

Size of Disturbance	Rate of Change in Disturbance Size	Output of Disturbance Model	Output of Feedforward Controller
0	0	0	0
0	0	0	0
0.00056	0	0	0
0.00056	0.00448	0	168
0.00056	0.00896	0	154.75
0.00056	0.01344	0	143.04
0.00056	0.01792	0	132.71
0.00056	0.0224	0	123.57
0.00056	0.02688	-1.1741	115.5
0.00056	0.03136	-2.2112	108.38

It is shown in Table 4.10, that the data available to be used to train the neuro-fuzzy logic controller is insufficient to create a logical relationship. The change at the controller output could not be related to other data changes thus producing a large training error for the neuro-fuzzy logic controller (unacceptable).

## **CHAPTER 5**

### **CONCLUSION AND RECOMMENDATIONS**

#### **5.1 Conclusion**

The aim of this project is to improve the performance of the neuro-fuzzy logic controller for the heat exchanger temperature control. The improvement is on the design of the controller for the heat exchanger so that the neuro-fuzzy logic controller can be operated for the whole operating range or set points of the heat exchanger.

Theoretical knowledge in heat exchanger, PID controller, neural network and neuro-fuzzy logic controller is emphasized and being the main scopes of the study in order to complete this project.

Based on the results, it can be concluded that the neuro-fuzzy logic controller has been improved and can perform well over several difference SP changes by training it with multiple set of training data. The comparison between the PID controller and the neuro-fuzzy logic controller shows that these controllers are close to each other in terms of performance.

## 5.2 Recommendation

Technically the neuro-fuzzy logic controller could be improved more by selecting the appropriate training set of data. In this project, four sets of training data have been used to train the neuro-fuzzy logic controller to replace the PID controller. Even though the performance is lesser than the PID controller if viewed from the CV overshoot and the decay ratio but still it performed under the acceptable range of CV overshoot ( $\leq 20\%$ ) and decay ratio ( $\leq 25\%$ ).

The feedforward controller actions theoretically differ with the feedback controller. A further studies need to be done to the feedforward controller in order to implement a neuro-fuzzy logic controller for the feedforward controller.

In the near future, if possible, the project could be extended by implementing both controllers in one process loop where these controllers works together to perform or to produce a better result.

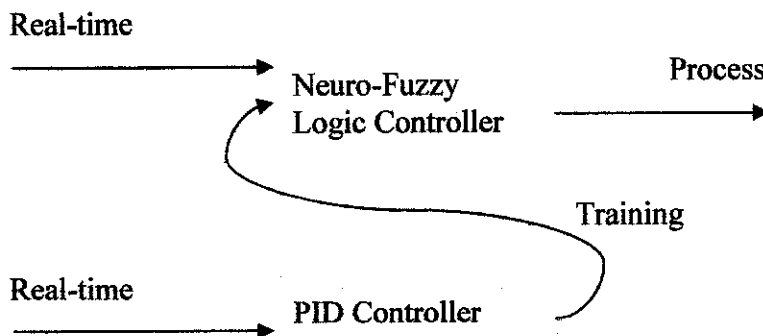


Figure 5.1: Proposed control strategy with two controllers working together

## REFERENCES

- [1] FUZZY LOGIC CONTROLLER FOR HEAT EXCHANGER TEMPERATURE CONTROL, *Mohd Faizal Ja'afar*, Electrical & Electronics Engineering, Universiti Teknologi Petronas, December 2004.
  
- [2] IMPROVEMENTS TO FUZZY LOGIC CONTROLLER FOR HEAT EXCHANGER, *Rozairi bin Saliman*, Electrical & Electronics Engineering, Universiti Teknologi Petronas, June 2006.
  
- [3] PROCESS CONTROL, *Designing Processes and Control System for Dynamic Performance*, Thomas E. Marlin, Mc Graw Hill 2<sup>nd</sup> Edition.
  
- [4] PLANT PROCESS CONTROL SYSTEM, *Lecture Materials*, Suhaila Badarol Hisham, Electrical & Electronics Engineering, Universiti Teknologi Petronas, January 2007.
  
- [5] HEAT EXCHANGERS,  
<http://www.me.wustl.edu/ME/labs/thermal/me372b5.htm>
  
- [6] FUZZY LOGIC, From Wikipedia, the free encyclopedia  
[http://en.wikipedia.org/wiki/Fuzzy\\_logic](http://en.wikipedia.org/wiki/Fuzzy_logic)
  
- [7] Definitions of Neural network  
[www.inproteomics.com/nwglosno.html](http://www.inproteomics.com/nwglosno.html)
  
- [8] PID CONTROLLER, From Wikipedia, the free encyclopedia  
[http://en.wikipedia.org/wiki/PID\\_controller](http://en.wikipedia.org/wiki/PID_controller)

# APPENDIX A

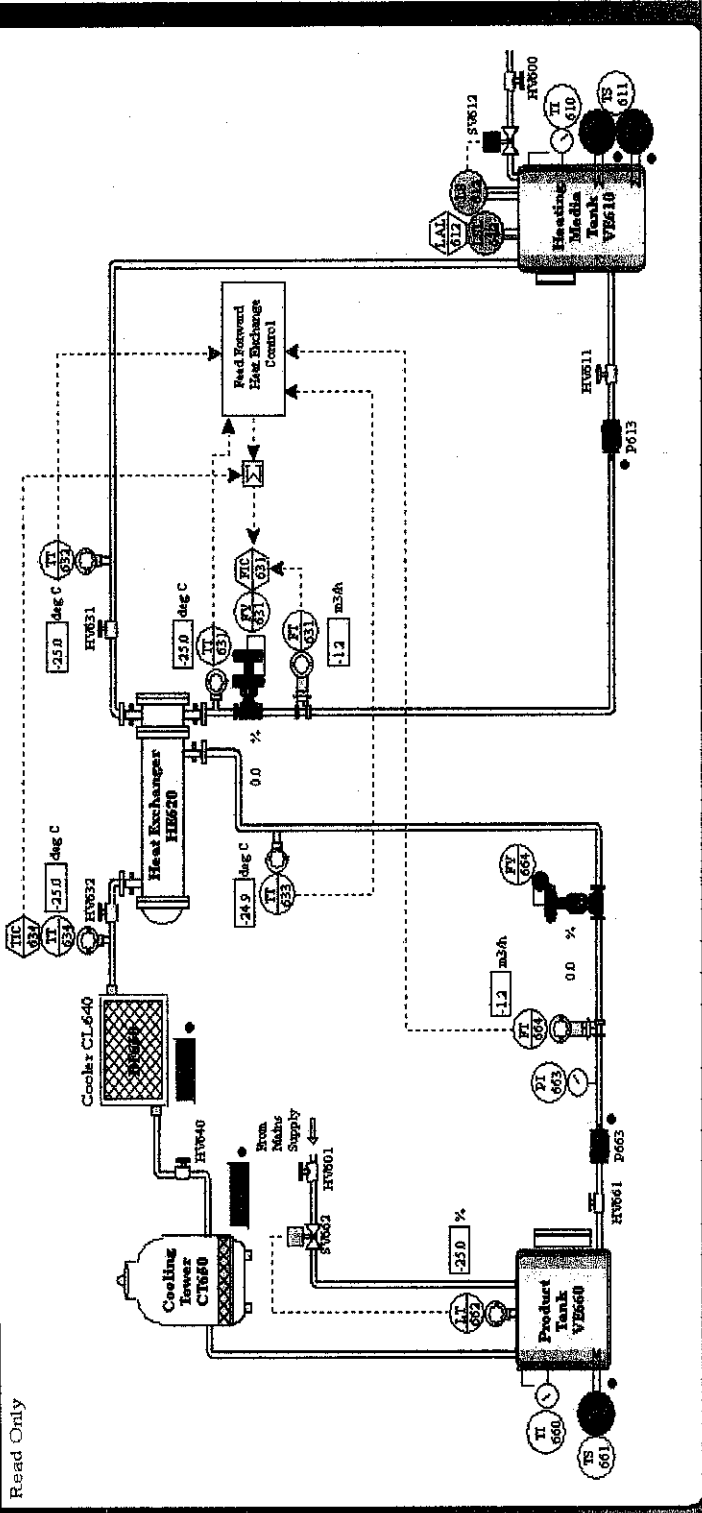
PC Automation S&B, Bhd

SIM 305 Pilot Plant

Experiment 7

Feedforward Temperature With Feedback Trim

19-Oct-2016 04:05:07 PM

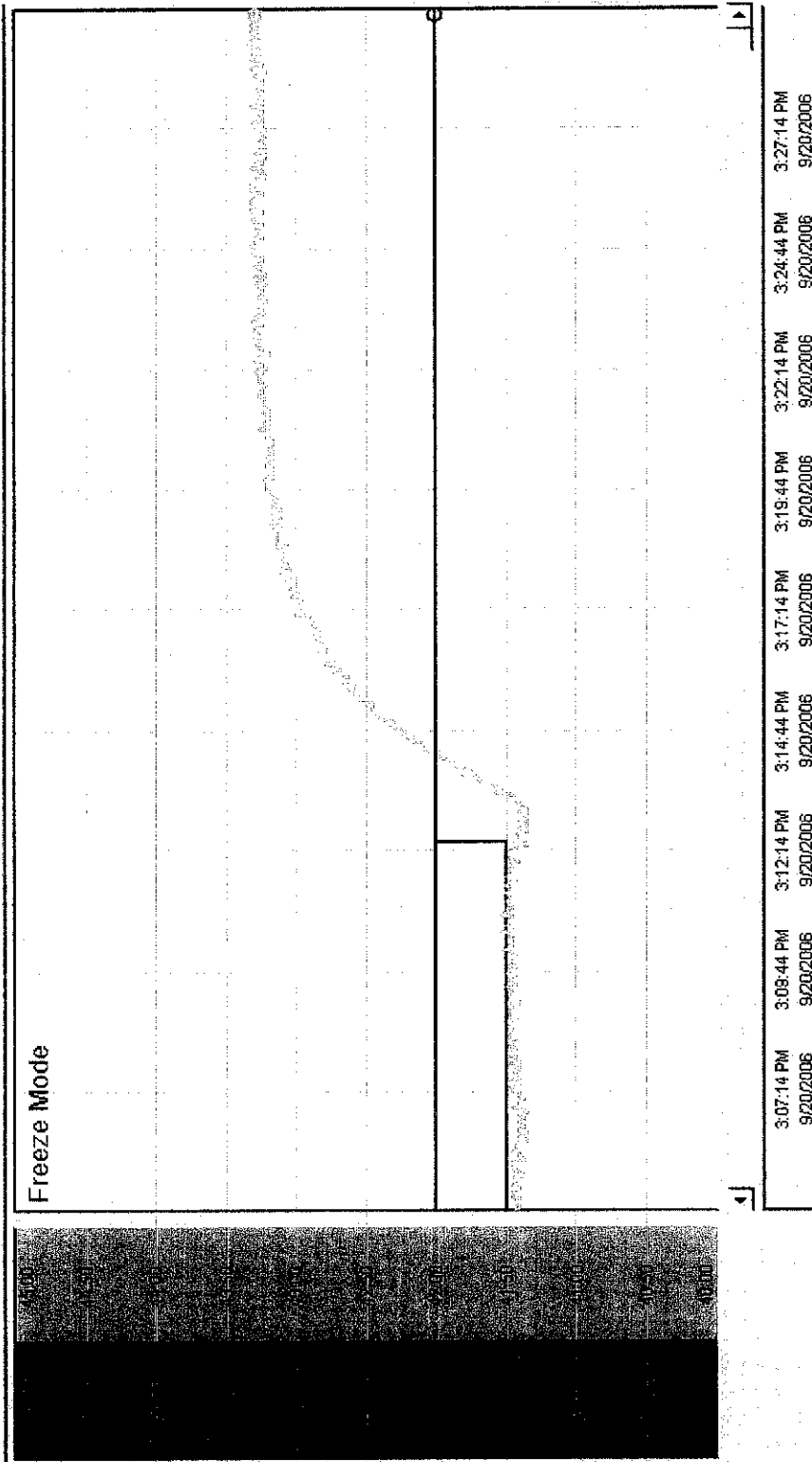


Time / Date	Description	Tag	Value	Priority	Type	Quality	Node	Comment	Event Time
3:58:56 PM	107 LO-Limit	FI651	0	500	LO	Good - Non			3:58:56 PM 10/19/16
Home	Overhaul				FF Trim Trend				Alarm

## OVERVIEW OF SIM305 PILOT PLANT 6: HEAT EXCHANGER

## APPENDIX B

This appendix shows the detail of determining the rate of difference for TF1 and TF2



Process Reaction Curve (PRC) obtained from the lab experiment

## APPENDIX B

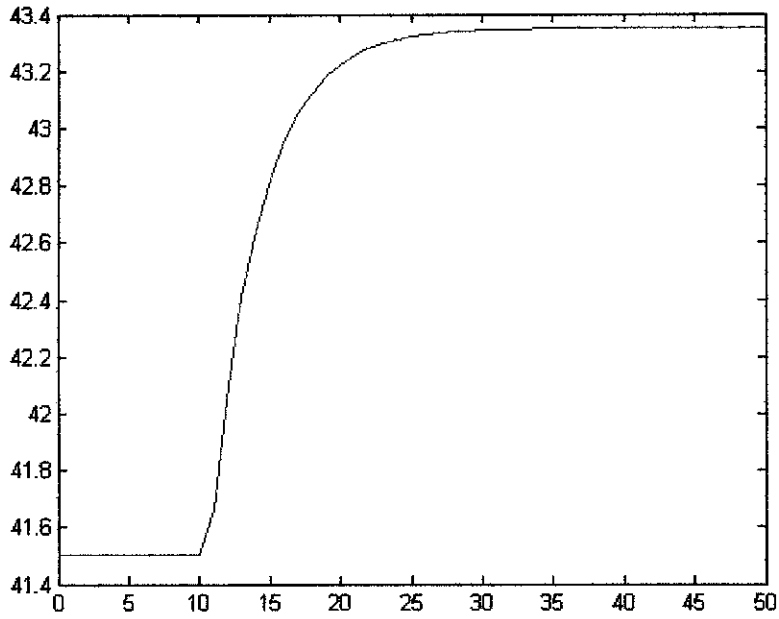
-The PRC in numerical data.

-These data is estimated from the curve because the actual data is corrupted.

41.41	41.41	41.65	42.10	42.32	42.50	42.75	42.85	42.95	43.35	43.10	43.17	43.22	43.22	43.20	43.35	43.35
41.35	41.41	41.65	42.10	42.32	42.50	42.75	42.85	42.95	43.35	43.10	43.17	43.22	43.22	43.20	43.35	43.35
41.35	41.45	41.65	42.10	42.32	42.50	42.75	42.85	42.95	43.35	43.12	43.17	43.22	43.22	43.20	43.35	43.35
41.35	41.45	41.65	42.10	42.32	42.50	42.75	42.85	42.95	43.35	43.12	43.17	43.22	43.22	43.20	43.35	43.35
41.35	41.45	41.65	42.10	42.32	42.50	42.75	42.85	42.95	43.35	43.12	43.10	43.22	43.22	43.25	43.35	43.35
41.35	41.45	41.80	42.10	42.32	42.50	42.75	42.85	42.95	43.10	43.12	43.10	43.22	43.22	43.25	43.35	43.35
41.35	41.45	41.80	42.10	42.32	42.50	42.75	42.85	42.95	43.10	43.12	43.10	43.22	43.20	43.25	43.35	43.35
41.35	41.45	41.80	42.20	42.32	42.50	42.75	42.85	42.95	43.10	43.12	43.10	43.22	43.18	43.25	43.35	43.35
41.35	41.45	41.80	42.30	42.39	42.50	42.82	42.85	42.95	43.10	43.12	43.10	43.22	43.18	43.25	43.35	43.35
41.35	41.45	41.80	42.30	42.39	42.70	42.82	42.85	42.95	43.35	43.12	43.10	43.22	43.18	43.25	43.35	43.35
41.35	41.50	41.80	42.30	42.39	42.70	42.82	42.85	42.95	43.35	43.12	43.10	43.22	43.18	43.25	43.35	43.35
41.35	41.50	41.80	42.20	42.39	42.70	42.82	42.85	42.95	43.35	43.12	43.17	43.22	43.18	43.25	43.35	
41.35	41.50	41.75	42.30	42.39	42.70	42.82	42.85	42.95	43.35	43.12	43.17	43.22	43.18	43.25	43.35	
41.35	41.50	41.75	42.20	42.39	42.70	42.82	42.90	43.05	43.10	43.12	43.17	43.22	43.18	43.25	43.35	
41.35	41.50	41.75	42.30	42.39	42.70	42.82	42.90	43.05	43.10	43.12	43.17	43.22	43.18	43.25	43.35	
41.35	41.50	41.75	42.20	42.39	42.70	42.82	43.00	43.05	43.10	43.16	43.10	43.17	43.18	43.25	43.35	
41.35	41.50	41.80	42.30	42.39	42.70	42.82	42.95	43.05	43.10	43.16	43.10	43.17	43.18	43.25	43.35	
41.35	41.50	41.80	42.30	42.39	42.70	42.82	42.95	43.05	43.10	43.16	43.10	43.17	43.18	43.25	43.35	
41.35	41.50	41.80	42.30	42.39	42.70	42.82	42.95	43.00	43.10	43.16	43.10	43.22	43.22	43.25	43.35	
41.35	41.50	41.80	42.30	42.44	42.70	42.82	42.95	43.00	43.10	43.10	43.10	43.22	43.22	43.25	43.35	
41.35	41.50	41.80	42.30	42.44	42.70	42.82	42.95	43.00	43.10	43.10	43.10	43.22	43.22	43.25	43.35	
41.35	41.50	41.80	42.30	42.44	42.73	42.82	42.95	43.00	43.10	43.10	43.10	43.22	43.22	43.25	43.35	
41.35	41.50	41.80	42.30	42.44	42.73	42.82	42.95	43.00	43.10	43.10	43.12	43.22	43.22	43.25	43.35	
41.35	41.50	41.85	42.30	42.44	42.73	42.82	42.95	43.00	43.10	43.10	43.12	43.22	43.22	43.25	43.35	
41.35	41.50	41.9	42.30	42.44	42.73	42.82	42.95	43.00	43.10	43.17	43.12	43.22	43.22	43.25	43.35	
41.35	41.50	42.00	42.21	42.44	42.73	42.82	42.95	43.00	43.10	43.17	43.12	43.22	43.22	43.25	43.35	
41.35	41.50	42.00	42.32	42.44	42.73	42.82	42.95	43.00	43.10	43.17	43.12	43.22	43.20	43.25	43.35	
41.41	41.50	42.00	42.32	42.55	42.73	42.82	42.95	43.00	43.10	43.17	43.12	43.22	43.20	43.25	43.35	
41.41	41.50	42.00	42.32	42.50	42.73	42.82	42.95	43.05	43.10	43.17	43.12	43.22	43.20	43.25	43.35	
41.41	41.50	42.00	42.32	42.50	42.75	42.82	42.95	43.05	43.10	43.17	43.12	43.22	43.20	43.25	43.35	
41.41	41.55	42.00	42.32	42.50	42.75	42.83	42.95	43.05	43.10	43.17	43.22	43.22	43.20	43.25	43.35	
41.41	41.55	42.00	42.32	42.50	42.75	42.85	42.95	43.05	43.10	43.17	43.22	43.22	43.20	43.25	43.35	
41.41	41.55	42.00	42.32	42.50	42.75	42.85	42.95	43.05	43.10	43.17	43.22	43.22	43.20	43.25	43.35	

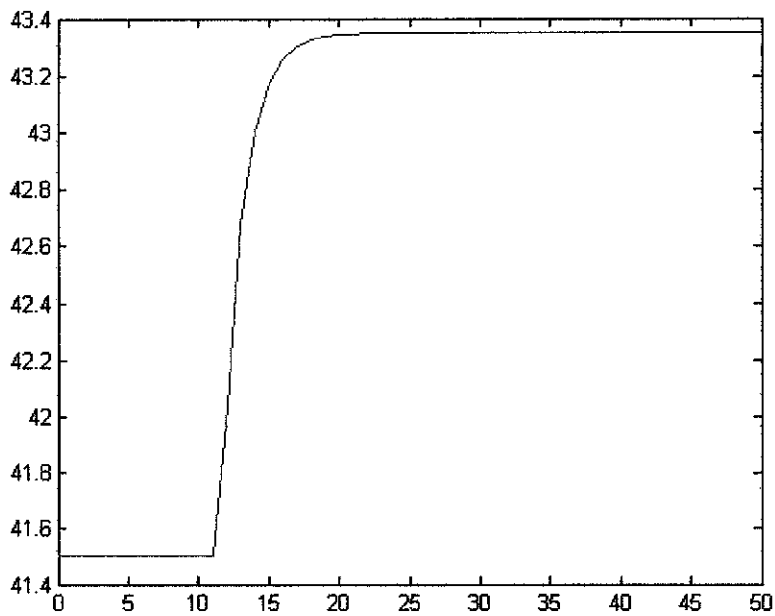
## APPENDIX B

### Open-loop Simulation Result for TF1 and TF2



**TF1**

$$\frac{Y(s)}{X(s)} = \frac{0.185e^{-42s}}{210s+1}$$



**TF2**

$$\frac{Y(s)}{X(s)} = \frac{0.185e^{-84s}}{90s+1}$$



## APPENDIX B

The numerical data from open-loop simulation for TF1 and TF2

TF 1		TF 2	
x-axis	y-axis	x-axis	y-axis
1	41.5	1	41.5
2	41.5	2	41.5
3	41.5	3	41.5
4	41.5	4	41.5
5	41.5	5	41.5
6	41.5	6	41.5
7	41.5	7	41.5
8	41.5	8	41.5
9	41.5	9	41.5
10	41.5	10	41.5
11	41.5	11	41.5
12	41.5	12	41.5
13	41.5	13	41.5
14	41.695	14	41.5
15	42.098	15	42.04
16	42.409	16	42.677
17	42.643	17	43.005
18	42.819	18	43.173
19	42.951	19	43.259
20	43.05	20	43.303
21	43.124	21	43.326
22	43.181	22	43.338
23	43.223	23	43.344
24	43.254	24	43.347
25	43.278	25	43.348
26	43.296	26	43.349
27	43.309	27	43.35
28	43.319	28	43.35
29	43.327	29	43.35
30	43.333	30	43.35
31	43.337	31	43.35
32	43.34	32	43.35
33	43.343	33	43.35
34	43.345	34	43.35
35	43.346	35	43.35
36	43.347	36	43.35
37	43.348	37	43.35
38	43.348	38	43.35
39	43.349	39	43.35
40	43.349	40	43.35
41	43.349	41	43.35
42	43.349	42	43.35
43	43.35	43	43.35
44	43.35	44	43.35
45	43.35	45	43.35
46	43.35	46	43.35
47	43.35	47	43.35
48	43.35	48	43.35
49	43.35	49	43.35
50	43.35	50	43.35

## APPENDIX B

Both transfer function (TF 1 and TF 2) is compared to the PRC and the absolute difference between them and the PRC are calculated. It is repeated for an increment of 1second until they reach steady – state.

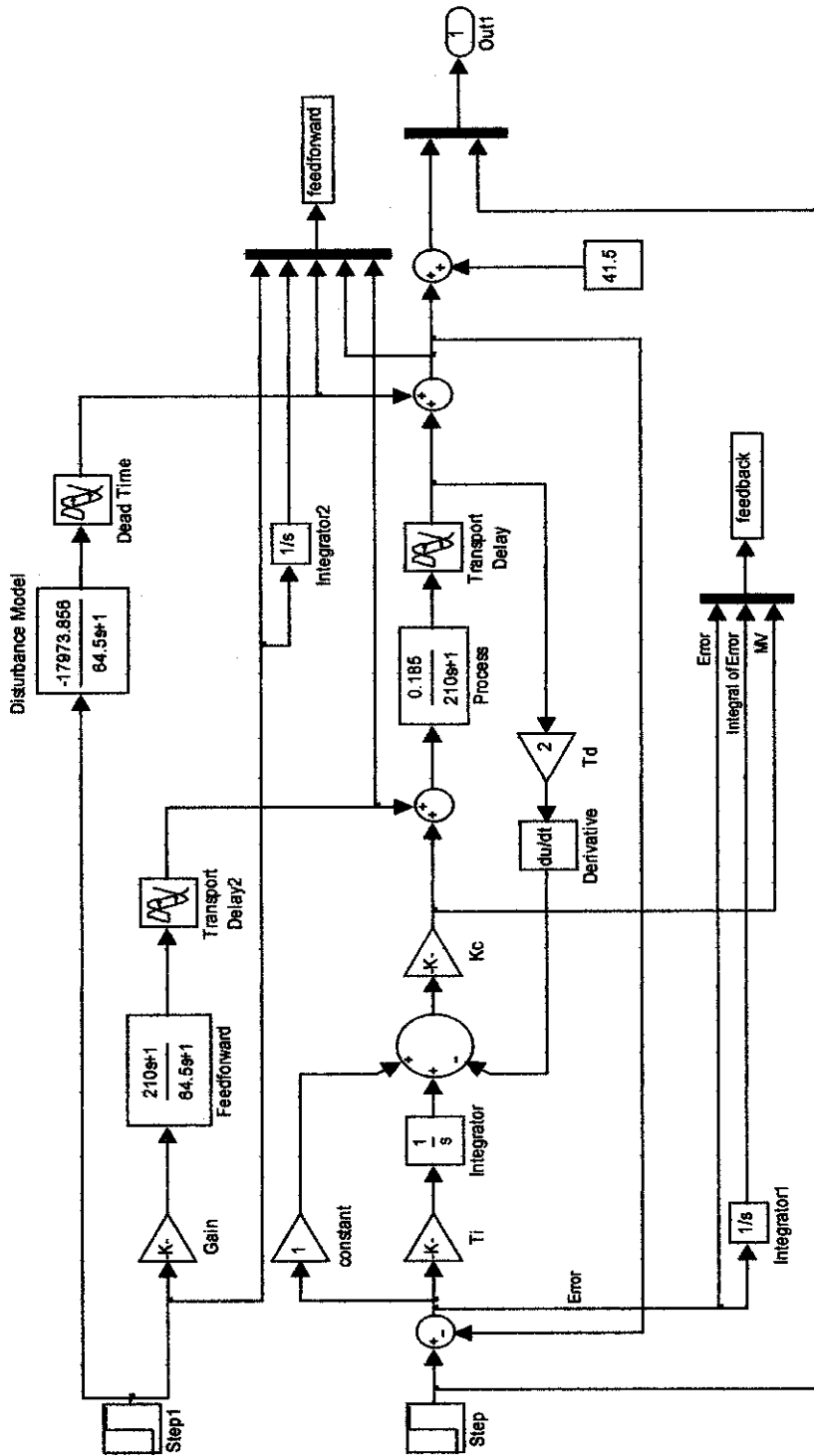
$$Diff (\%) = \frac{\sum [temp(TF) - temp(PRC)]}{Area(PRC)} \times 100\%$$

$$\begin{aligned} \text{TF 1} \quad \text{Percentage diff (\%)} &= 364.55 / 712.5 \\ &= \underline{51.16\%} \end{aligned}$$

$$\begin{aligned} \text{TF 2} \quad \text{Percentage diff (\%)} &= 366.56 / 712.5 \\ &= \underline{51.46\%} \end{aligned}$$

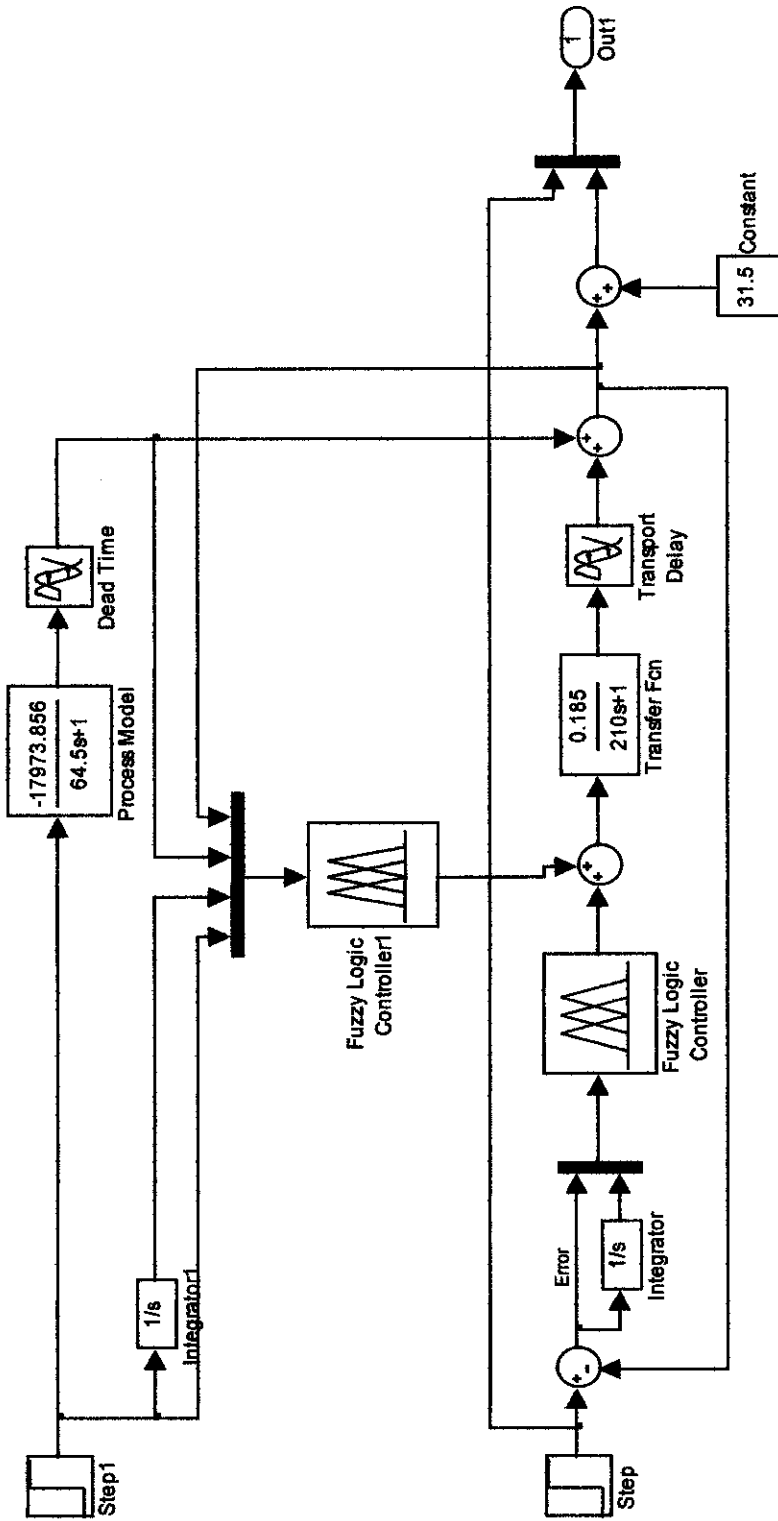
## APPENDIX C

The simulation block diagram (i) PID controller + feedforward controller and (ii) neuro-fuzzy logic controller



PID controller + feedforward controller

# APPENDIX C



Neuro-fuzzy logic controller