Project Report

on

PID Tuning of a Thermoregulation Device

at

CSIR – Central Scientific Instruments Organisation, Chandigarh

CSIR-CSIO

Dr. B. R. Ambedkar National Institute of Technology, Jalandhar

(Department of Instrumentation and Control engineering)

SUBMITTED TO : -

Dr.Sanjeev Soni

Senior Scientist

Biomedical Instrumentation

SUBMITTED BY : -

Arekh Tiwari

15106013

ACKNOWLEDGEMENT

It is fortune to find opportunity to express my deep gratitude to all people who helped me with their guidance and assistance for making this training happen. Their contribution has been valued.

I am thankful to Dr. Sanjeev Soni for giving me the opportunity to work under his guidance and complete this project successfully.

I would also like to thank Dr. Dinesh Pankaj who was also involved in the development of this project. Without his participation and input, the project would not have been completed.

Then, I would like to extend my heart fill gratitude and respect to my parents, who have always a tremendous source of encouragement & inspiring me. I would like to give my heartiest thanks to all faculty members for giving me the precious time incessant, encouragement & guidance to make this training success. They have been part in shaping my opinion about task at hand by holding our fruitful and result oriented discussion supported by constructive criticism.

I am also thankful to all those near and dear ones, knowing or unknowing who have helped me out in any way having no names here.

Arekh Tiwari

Roll No. :15106013

Branch : Instrumentation and Control Engineering

Dr.B.R. Ambedkar National Institute of Technology, Jalandhar

Certificate

This is to certify that the Project Report Entitled "**PID Tuning of a Thermoregulation Device**" which is submitted by **Mr. Arekh Tiwari** , Roll No. **15106013** student of **Dr. B. R. Ambedkar National Institute of Technology , Jalandhar** is an authentic work carried out by him at **CSIR – CSIO , Chandigarh** during the period **11th June 2018** to **27th July 2018** under my guidance.

Dr . Sanjeev Soni 27/7/18

Senior Scientist

Biomedical Instrumentation

CSIR – CSIO , Chandigarh

3

ABSTRACT

The aim of the work concentrates around the tuning of a thermoregulaion device. However, this project carries small part in the development of the device.

The work shows the variation of Response Characteristics due to change in PID parameters Kp, Ki & Kd and proposed the scheme to implement tuned PID parameters to control over the temperature during entire operation and to raise the patient's body temperature to the normal level during hypothermia and to maintain it to the same level during the entire perioperative period.. Low body temperature causes complications like coma or even death. This device maintains the temperature of the patient body by creating a warm environment surrounding the patient body by a pleasantly heated air through a surgical blanket. Therefore to produce the air at a precise temperature level from the device, the device needs to be well designed and analyzed for its operation. Analysis of such device is presented here. Simulated temperatures and tuning schemes are compared with a suitable experiment. Temperature vs Time characteristics for various PID Coefficients are compared and analysed. The PID coefficients which are the most suitable for Thermoregulation device are selected.

Contents

Chapter 1

Introduction

The human body temperature is a vital parameter, just as vital as respiration rate or blood pressure or pulse rate. The human body is considered as a type of small furnace which generates heat every moment. The heat generated is responsible for the rise in temperature of the human body. Nose, ear, mouth, under the arm and rectal areas are the sites in the human body used for the measurement of the body temperature. Normally, the average human body temperature lies around 37°C but, may vary from person-to-person. A person's body temperature changes at every moment of the day, although these changes may only be very small. Generally, body temperature depends upon age, sex, exertion, time of the day, place, physical activity performed by the person and some type of foods and drinks.

It is obvious that sometimes body temperature may rise and sometimes it may decrease, both these conditions are dangerous for the human body. However, the human body is always trying to maintain this temperature variation. The process of maintaining temperature is called thermoregulation and system which maintains this thermal variation in the body is called as the thermoregulatory system.

1.1 Thermoregulatory system of human body

The term thermoregulatory consist of two distinct words- thermo refers to temperature and regulatory relates with regulation. Therefore, a system which is capable of regulating the temperature in the human body is known as the thermoregulatory system. The thermoregulatory system of the human body is formed by combining a number of parts of the body, each having their own significance. Hypothalamus, a part of the brain and the centre of the thermoregulatory system, collects data from other parts of the brain, spinal cord, tissues and peripheral thermal sensors in the skin. This information is processed in the hypothalamus itself and then the processed information is transported through the central nervous system to the organism. After that the organism starts to regulate the temperature of the body in proper direction i.e.,

if the body temperature is greater than the standard range then the organs will try to make the temperature lower by the process of perspiration and vasodilation and if the body temperature becomes lower than the normal then the organism calls shivering (thermo genesis) and vasoconstriction to increase the body temperature. Ideally, the thermoregulatory system of the body regulates the core body temperature to about 37°C +/-0.5°C - referred to as normothermia .However, for the healthy functioning of the body, the core body temperature must lie within the range of +/- 0.2°C from the normal value Figure 1.1: Graphic used to show thermal compartments of the human body.

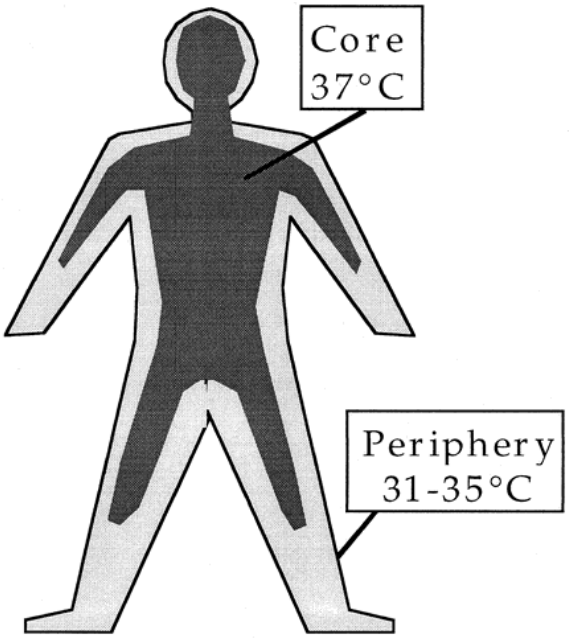


Fig 1.1 *Graphic used to show thermal compartments of human body*

The thermoregulatory system maintains only the core temperature of the body i.e., there are some other temperature zones also present in the body. Broadly, the human body can be divided into two thermal compartments, namely the core thermal compartment and the peripheral thermal compartment .Figure 1.1 shows the average temperature of both these thermal compartments. The core thermal compartment mainly consists of the trunk and the head. The temperature in this zone remains relatively uniform at around 37°C. On the other hand, the peripheral thermal compartment consists of the skin and the outermost tissues of the head and trunk. The temperature in this compartment is not as uniform as in the core compartment. It

varies between 31°C – 35°C. Usually, the temperature in the peripheral thermal compartment is 2- 5°C less than the core temperature in moderate environments ,but this difference may change during extreme thermal or physiological circumstances.

1.2 Thermoregulatory system as a closed loop control system

As discussed earlier, the thermoregulatory system maintains the temperature of the body in a very narrow zone of +/- 0.5°C. Maintaining such a close temperature range is a very difficult task, but the human body executes this type of work efficiently. The relation of the thermoregulatory system with the control system is discussed.

Figure 1.2: Block diagram representation simple closed loop control system.

In **closed loop control systems**, output is fed back to the input. So, the control action is dependent on the desired output.

The error detector produces an error signal, which is the difference between the input and the feedback signal. This feedback signal is obtained from the block (feedback elements) by considering the output of the overall system as an input to this block. Instead of the direct input, the error signal is applied as an input to a controller.

So, the controller produces an actuating signal which controls the plant. In this combination, the output of the control system is adjusted automatically till we get the desired response. Hence, the closed loop control systems are also called the automatic control systems.

In this project the **Input** is the reference temperature i.e. the temperature at which you want to set the patient's body.

The **Controller** is Arduino UNO which will send an actuating Signal to the Plant.

The **Plant** is the Human Body/Patients body whose temperature is to be controlled.

Ouput is the temperature of the human body which is sensed by a Temperature Sensor and fed back to the error detector which generates an error which acts as an input to the controller .

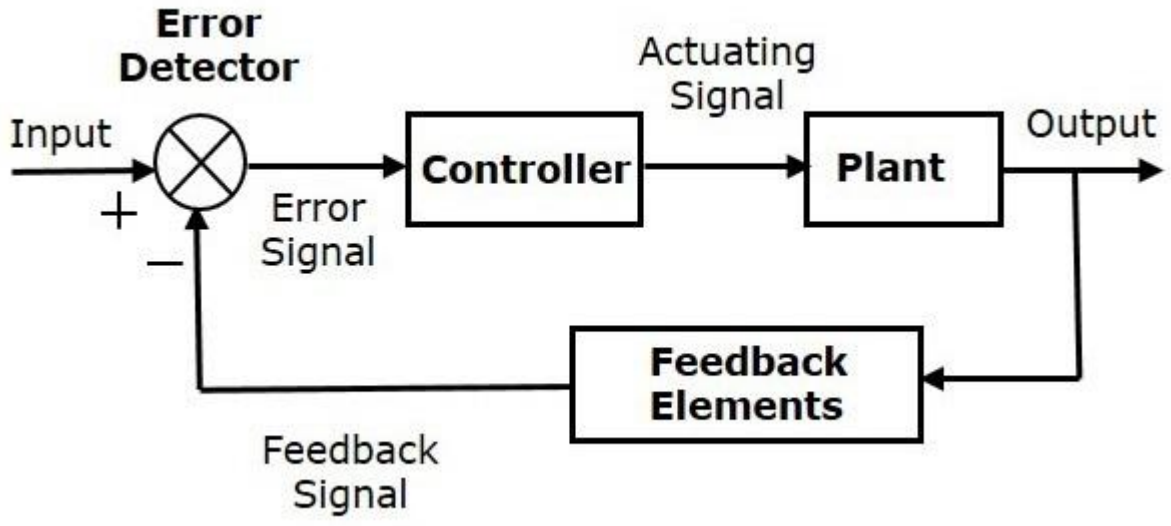


Fig 1.2 *Closed Loop Control System*

1.3 Intro to Thermoregulation Device

The term perioperative period refers to the period during a surgical operation. Normally, the perioperative period consists of three distinguished phases of surgery, namely preoperative - before surgery, perioperative - during surgery and postoperative after surgery.

According to medical definition, perioperative period is defined as the period between patient arrival for surgery to the discharge of the patient" Despite the fact that during a surgery the body experiences a number of cuts and incisions which cause pain to the patient, yet surgeries are frequently performed. Surgeries are often required for the treatment of diseases or for organ transplants.

Sometimes, surgeries are even called for the improvement in the body functionality and sometimes it is performed only for finding the problem in the body. There is always pain associated with surgeries, and this pain can make the patient anxious. The anxiety of the patient leads to a laborious surgery. Hence, it is necessary to make the patient unconscious during surgery such that the patient neither suffers from the extreme pain of cuts and incisions nor fall into anxiety. Therefore, surgeons introduce anaesthetic drugs to the patients in order to make them indifferent towards the surgical procedure, ensuring that the patient remains unaware of the cuts and incisions. The

condition in which patient becomes unconscious due to anaesthetic drugs is known as anaesthesia. Although anaesthesia is forced for a good cause i.e., making the patient insensitive to the surgical complications, it causes some pernicious effects on the patient's body.

One such effect is on the thermoregulatory system of the body such that the thermoregulatory system starts malfunctioning. It is discussed earlier that the suppression of thermoregulatory mechanism either results in the decrease in temperature of the body called hypothermia or increase in temperature of the body called hyperthermia .Since both conditions are problematic they must be avoided.

Hence, to overcome the aw in the thermoregulatory system of the body, an external device is required to regulate the body temperature. This device is called a **Thermoregulation device**.

1.4 Need of PID Control and Tuning

Now the question comes why do we need PID Control. When compared to other control algorithms, like model-based or matrix based ones the PID has the following advantages :-

1 - Easier to implement (just a simple equation).

2 - Uses lower resources.

3 - More robust to tuning mismatches (very important point, generally forgotten).

4 - Easier to tune by simple trial and error (robustness also helps a lot the non-experienced).

5 - Better response to unmeasured disturbances. Model-based controllers recover from unmeasured disturbances with only an integral type of action, while PID has also the proportional and derivative action that immediately act on an unknown disturbance.

Again, the choice of the value of the P, I and D parameters is very much process dependent. As a result, thorough knowledge about the plant dynamics is important for selection of these parameters. In most of the cases, it is difficult to obtain the exact

mathematical model of the plant. So, we have to rely on the experimentation for finding out the optimum settings of the controller for a particular process. The process of experimentation for obtaining the optimum values of the controller parameters with respect to a particular process is known as **Controller Tuning**. It is needless to say, that controller tuning is very much process dependent and any improper selection of the controller settings may lead to instability, or deterioration of the performance of the closed loop system.

Chapter 2

Theory

This chapter mainly introduces the devices, the basic structure of the control system and the PID tuning. Therefore, it provides the basic theory of this study for readers. Next chapter will explain the whole process and test which based on these theory knowledge.

2.1 PID Control

A **proportional–integral–derivative controller** (**PID controller**) is a control loop feedback mechanism (controller) commonly used in industrial control systems. A PID controller continuously calculates an error value as the difference between a desired set point and a measured process variable and applies a correction based on proportional, integral, and derivative terms (sometimes denoted *P*, *I*, and *D* respectively) which give their name to the controller type.

The PID control scheme is named after its three correcting terms, whose sum constitutes the manipulated variable (MV). The proportional, integral, and derivative terms are summed to calculate the output of the PID controller. Defining u(t) as the controller output, the final form of the PID algorithm is

$$u(t) = \overbrace{K_p e(t)}^{\text{Proportional}} + \overbrace{K_i \int_0^t e(\tau)\,d\tau}^{\text{Integral}} + \overbrace{K_d \frac{d}{dt} e(t)}^{\text{Derivative}}$$

where ,

K_p is the proportional gain, a tuning parameter,

K_i is the integral gain, a tuning parameter,

K_d is the derivative gain, a tuning parameter,

***e(t)* = SP - PV(t)** is the error (SP is the setpoint, and PV(*t*) is the process variable),

t is the time or instantaneous time (the present),

τ is the variable of integration (takes on values from time 0 to the present **t**).

P-Control (Proportional Control)

The proportional term produces an output value that is proportional to the current error value. The proportional response can be adjusted by multiplying the error by a constant K_p, called the proportional gain constant.

The proportional term is given by

$$\mathbf{P_{out}} = \mathbf{K_p}\, e(t)$$

A high proportional gain results in a large change in the output for a given change in the error. If the proportional gain is too high, the system can become unstable (see the section on loop tuning). In contrast, a small gain results in a small output response to a large input error, and a less responsive or less sensitive controller. If the proportional gain is too low, the control action may be too small when responding to system disturbances. Tuning theory and industrial practice indicate that the proportional term should contribute the bulk of the output change.

I-Control (Integral Control)

The contribution from the integral term is proportional to both the magnitude of the error and the duration of the error. The integral in a PID controller is the sum of the instantaneous error over time and gives the accumulated offset that should have been corrected previously. The accumulated error is then multiplied by the integral gain (K_i) and added to the controller output.

The integral term is given by

$$K_i \int_0^t e\,dt$$

The integral term accelerates the movement of the process towards setpoint and eliminates the residual steady-state error that occurs with a pure proportional

controller. However, since the integral term responds to accumulated errors from the past, it can cause the present value to overshoot the setpoint value .

D Control (Differential Control)

The derivative of the process error is calculated by determining the slope of the error over time and multiplying this rate of change by the derivative gain K_d. The magnitude of the contribution of the derivative term to the overall control action is termed the derivative gain, K_d.

The derivative term is given by

$$K_d \frac{d}{dt} e$$

Derivative action predicts system behaviour and thus improves settling time and stability of the system. An ideal derivative is not causal, so that implementations of PID controllers include an additional low-pass filtering for the derivative term to limit the high-frequency gain and noise. Derivative action is seldom used in practice though – by one estimate in only 25% of deployed controllers – because of its variable impact on system stability in real-world applications.

The PID Control

PID control is namely the combination of proportional control, integral control and differential control, thus integrating the advantages of such three kinds of controllers. In practical application, there is no need to use all these three parts, but only proportional control unit is indispensable. For the PID controller, the output is:
Where Kp = Proportional gain, Ki= Integral gain, Kd = Derivative gain.
Finally we can get a complete PID control.

$$u(t) = \underbrace{K_p e(t)}_{\text{Proportional}} + \underbrace{K_i \int_0^t e(\tau)d\tau}_{\text{Integral}} + \underbrace{K_d \frac{d}{dt} e(t)}_{\text{Derivative}}$$

PID control actually means setting these three parameters, namely, Kp, Ti and Td, in order to get applicable output value to control the system. The specific details on how to set them are different based on different situations. Currently, PID is not only widely applied but also rapidly developed. The intelligent controllers which can self-tune these three parameters have been massively invented .

After the combination between PID and digital controllers such as computer, the design method of digital PID has also emerged, whose specific principle still follows the traditional ones.

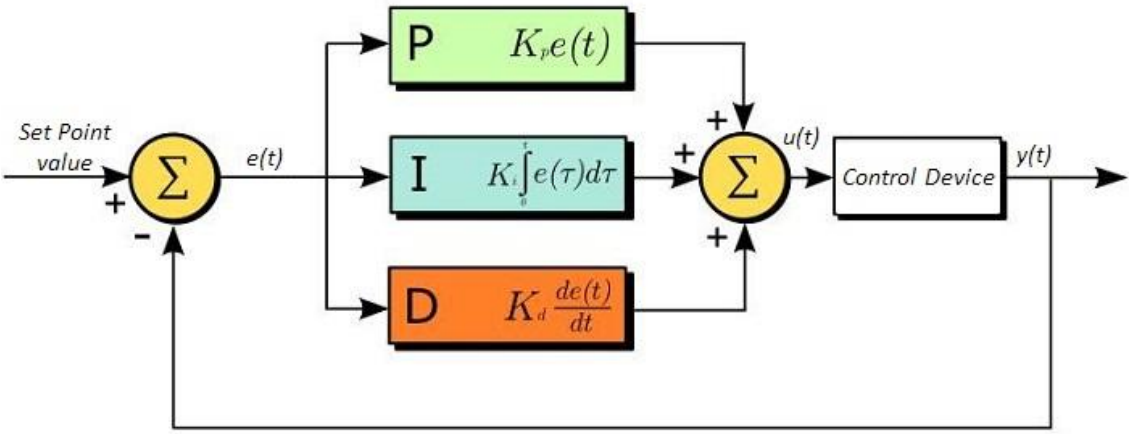


Figure 2.1

2.2 PID Tuning Algorithms

2.2.1 Manual PID Tuning

If the system must remain online, one tuning method is to first set Ki and Kd values to zero. Increase the Kp until the output of the loop oscillates, then the Kp should be set to approximately half of that value for a "quarter amplitude decay" type response. Then increase Ki until any offset is corrected in sufficient time for the process. However, too much Ki will cause instability. Finally, increase Kd, if required, until

the loop is acceptably quick to reach its reference after a load disturbance. However, too much Kd will cause excessive response and overshoot. A fast PID loop tuning usually overshoots slightly to reach the setpoint more quickly; however, some systems cannot accept overshoot, in which case an overdamped closed-loop system is required, which will require a Kp setting significantly less than half that of the Kp setting that was causing oscillation.

Parameter	Rise time	Overshoot	Settling time	Steady-state error	Stability
Kp	Decrease	Increase	Small change	Decrease	Degrade
Ki	Decrease	Increase	Increase	Eliminate	Degrade
Kd	Minor change	Decrease	Decrease	No effect in theory	Improve if Kd small

Table 2.1 *Effects of increasing a parameter independently*

2.2.2 Ziegler Nichols Tuning Method

The time-honoured Ziegler-Nichols tuning rule (*"Z-N rule"*), as introduced in the 1940s, had a large impact in making PID feedback controls acceptable to control engineers. PID was known, but applied only reluctantly because of stability concerns. With the Ziegler-Nichols rule, engineers finally had a practical and systematic way of tuning PID loops for improved performance. Never mind that the rule was based on science fiction. After taking just a few basic measurements of actual system response, the tuning rule confidently recommends the PID gains to use.

The Ziegler-Nichols rule is a heuristic PID tuning rule that attempts to produce good values for the three PID gain parameters:

1. *Kp* - the controller path gain

2. *Ti* - the controller's integrator time constant
3. *Td* - the controller's derivative time constant

given two measured feedback loop parameters derived from measurements:

1. the period *Tu* of the oscillation frequency at the stability limit
2. the gain margin *Ku* for loop stability

Steps for tuning PID Parameters using Ziegler Nichols Tuning Method are as follows:-

1. First, note whether the required proportional control gain is positive or negative. To do so, step the input u up (increased) a little, under manual control, to see if the resulting steady state value of the process output has also moved up (increased). If so, then the steady-state process gain is positive and the required Proportional control gain, K_c, has to be positive as well.
2. Turn the controller to P-only mode, i.e. turn both the Integral and Derivative modes off.
3. Turn the controller gain, K_c, up slowly (more positive if K_c was decided to be so in step 1, otherwise more negative if K_c was found to be negative in step 1) and observe the output response. Note that this requires changing K_c in step increments and waiting for a steady state in the output, before another change in K_c is implemented.
4. When a value of K_c results in a sustained periodic oscillation in the output (or close to it), mark this critical value of K_c as K_u, the ultimate gain. Also, measure the period of oscillation, P_u, referred to as the ultimate period. (Hint: for the system A in the PID simulator, K_u should be around 0.7 and 0.8)
5. Using the values of the ultimate gain, K_u, and the ultimate period, P_u, Ziegler and Nichols prescribes the following values for K_c, t_I and t_D, depending on which type of controller is desired:

Controller	K_p	T_i	T_d
P	0.5 K_{cr}	Infinity	0
PI	0.45 K_{cr}	$P_{cr}/1.2$	0
PID	0.6 K_{cr}	$P_{cr}/2$	0.125 P_{cr}

Table 2.2 *Ziegler Nichols Tuning Chart*

As an alternative to the table above, another set of tuning values have been determined by Tyreus and Luyblen for PI and PID, often called the TLC tuning rules. These values tend to reduce oscillatory effects and improves robustness.

Type of Controller	K_p	T_i	T_d
P	$K_u / 3.2$	$2.2P_u$	-
PID	$K_u / 2.2$	$2.2P_u$	$P_u / 6.3$

Table 2.3 *Tyreus-Luyben Tuning Chart*

Assuming the control loop is linear and the final control element is in good working order, you can continue with tuning the controller. The Ziegler-Nichols open-loop tuning rules use three process characteristics: process gain, dead time, and time constant. These are determined by doing a step test and analyzing the results

2.2.3 Ziegler Nichols Open Loop Tuning.

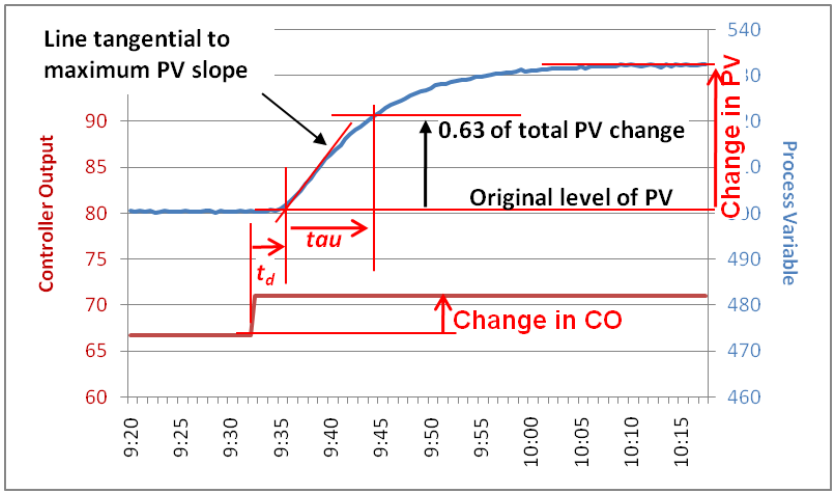


Fig 2.2 *Open Loop Response of System*

It uses ¼ decay ratio as design criterion. The Ziegler Nichols Open loop tuning criteria performs well when tau > 2td (lag dominant). It performs very poorly when td > 2tau (dead time dominant).

Fast recovery from disturbance but leads to oscillatory response.

Procedure:-

The process dynamics is modelled by a first order plus dead time model as given below ,

$$Y(s) = \left[\frac{Ke^{-st_0}}{\tau s + 1}\right] X(s)$$

After calculating the dead time and time constant either from the open loop response or from the transfer function. Use the below Table 2.3 i.e. the Ziegler Nichols Open Loop Tuning Chart to calculate the values of the PID parameters.

Controller	K_c	τ_i	τ_d
P	$\frac{1}{K_m}\frac{\tau_m}{t_d}$	-	-
PI	$\frac{0.9}{K_m}\frac{\tau_m}{t_d}$	$\frac{t_d}{0.3}$	-
PID	$\frac{1.2}{K_m}\frac{\tau_m}{t_d}$	$2t_d$	$0.5t_d$

Table 2.3 *Ziegler Nichols open loop Tuning Chart*

2.2.4 The C-H-R Method of PID Tuning

This method that has proposed by Chien, Hrones and Reswich [1] is a modification of open loop Ziegler and Nichols method. They proposed to use “quickest response

without overshoot" or "quickest response with 20% overshoot" as design criterion. They also made the important observation that tuning for set point responses and load disturbance responses are different. To tune the controller according to the C-H-R method the parameters of first order plus dead time model are determined in the same manner of the Z-N method. The controller parameters can then be determined from the Tables.6 and 7. The tuning rules based on the 20% overshoot design criterion are quite similar to the Z-N method. However when the 0% overshoot criteria is used, the gain and the derivative time are smaller and the integral time is larger. This means that the proportional action and the integral action, as well as the derivative action, are smaller.

Overshoot	0%			20%		
Controller Type	K_c	τ_I	τ_D	K_c	τ_I	τ_D
P	$\frac{0.3}{K_m}\frac{\tau_m}{d}$	—	—	$\frac{0.7}{K_m}\frac{\tau_m}{d}$	—	—
PI	$\frac{0.6}{K_m}\frac{\tau_m}{d}$	4d	—	$\frac{0.7}{K_m}\frac{\tau_m}{d}$	2.3d	—
PID	$\frac{0.95}{K_m}\frac{\tau_m}{d}$	2.4d	0.42d	$\frac{1.2}{K_m}\frac{\tau_m}{d}$	2d	0.42d

Table 2.4 *Tuning Relations for C-H-R Model*

2.2.5 Cohen Coon Method

In this method the process reaction curve is obtained first, by an open loop test as shown in Figure 1, and then the process dynamics is approximated by a first order plus dead time model, with following parameters:

$$\tau_m = \frac{3}{2}(t_2 - t_1)$$

$$d_m = \tau_2 - \tau_m$$

Where ,

t_1 = time at which $\Delta C = 0.283\ \Delta C_s$

t_1 = time at which $\Delta C = 0.632\ \Delta C_s$

C = the plant output. This method that proposed by Dr C. L. Smith provides a good approximation to process reaction curve by first order plus dead time model After determining of three parameters of km ,tm and d, the controller parameters can be obtained, using Cohen-Coon relations given in Table 2.5. These relations were developed empirically to provide closed loop response with a ¼ decay ratio.

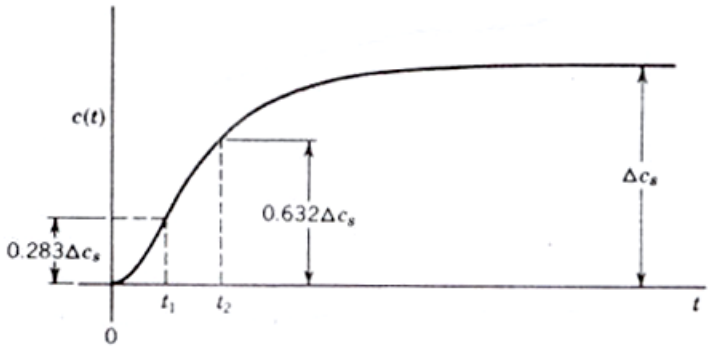


Fig.2.3 *Estimation of Parameters of FOPDT model*

Controller Type	k_c	τ_I	τ_D
P	$\frac{1}{K_m}\frac{\tau_m}{d}(1+\frac{d}{3\tau_m})$	-	-
PI	$\frac{1}{K_m}\frac{\tau_m}{d}(\frac{9}{10}+\frac{d}{12\tau_m})$	$d\frac{30+3d_m/\tau_m}{9+20d_m/\tau_m}$	-
PD	$\frac{1}{K_m}\frac{\tau_m}{d}(\frac{5}{4}+\frac{d}{6\tau_m})$	-	$d\frac{6-2d/\tau_m}{22+3d/\tau_m}$
PID	$\frac{1}{K_m}\frac{\tau_m}{d}(\frac{4}{3}+\frac{d}{4\tau_m})$	$d\frac{32+6d/\tau_m}{13+8d/\tau_m}$	$d\frac{4}{11+2d/\tau_m}$

Table 2.5 *Cohen Coon Controller Settings*

2.2.6 Tuning Method Using Genetic Algorithm

In this method we define a fitness function of the plant which is usually the performance index that considers the entire closed loop response.

Some of such indexes are as below

1. Integral of the absolute value of the error (IAE)

$$\mathbf{IAE}=\int_0^\infty |e(t)|.dt$$

2. Integral of the square value of the error (ISE)

$$\mathbf{ISE}=\int_0^\infty e^2(t).dt$$

3. Integral of the time weighted absolute value of the error (ITAE)

$$\mathbf{ITAE}=\int_0^\infty t\,|e(t)|\,dt$$

4. Integral of the time weighted square of the error (ITSE)

$$\mathbf{ITSE}=\int_0^\infty t.e^2(t).dt$$

The fitness function which is defined is optimised using Genetic Algorithm .

A Genetic Algorithm (GA) is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA). Genetic algorithms are commonly used to generate high-quality solutions to optimization and search problems by relying on bio-inspired operators such as mutation, crossover and selection.

2.2.7 Skogestad's Tuning Algorithm

The design principle of Skogestad's method is as follows. The control system tracking transfer function T (s), which is the transfer function from the reference or setpoint to the (filtered) process measurement, is specified as a first order transfer function with time delay:

$$T(s)=\frac{y_{mf}(s)}{y_{m_{SP}}(s)}=\frac{1}{T_C s+1}e^{-\tau s}$$

where T_C is the time constant of the control system which the user must specify, and τ is the process time delay which is given by the process model (the method can however be used for processes without time delay, too). Figure A.1 shows the response in y_{mf} after a step in the setpoint y_{mSP} for (A.1). (This response will be

achieved only approximately because Skogestad's method is based on some simplifying assumptions.)

Skogestad's Tuning formulas are given in Table 2.6.

$H(s)$	K_p	T_i	T_d
$\frac{K}{s}e^{-\tau s}$	$\frac{1}{K(T_C+\tau)}$	$k_1\,(T_C+\tau)$	0
$\frac{K}{Ts+1}e^{-\tau s}$	$\frac{T}{K(T_C+\tau)}$	$\min\left[T,\ k_1\,(T_C+\tau)\right]$	0
$\frac{K}{(Ts+1)s}e^{-\tau s}$	$\frac{1}{K(T_C+\tau)}$	$k_1\,(T_C+\tau)$	T
$\frac{K}{(T_1s+1)(T_2s+1)}e^{-\tau s}$	$\frac{T_1}{K(T_C+\tau)}$	$\min\left[T_1,\ k_1\,(T_C+\tau)\right]$	T_2
$\frac{K}{s^2}e^{-\tau s}$	$\frac{1}{4K(T_C+\tau)^2}$	$4\,(T_C+\tau)$	$4\,(T_C+\tau)$

Table 2.6 *Skogestad's Tuning Rules*

Note that the tuning formulas are for the series PID , for parallel PID parameters use the following conversion formulae,

$$K_{p_p} = K_{p_s}\left(1+\frac{T_{d_s}}{T_{i_s}}\right)$$

$$T_{i_p} = T_{i_s}\left(1+\frac{T_{d_s}}{T_{i_s}}\right)$$

$$T_{d_p} = T_{d_s}\frac{1}{1+\frac{T_{d_s}}{T_{i_s}}}$$

2.2.8 Relay Tuning of PID Parameters

When you discuss loop tuning with instrument and control engineers, conversation soon turns to the Zeigler-Nichols (ZN) ultimate oscillation method. So given the tedious and possibly dangerous plant trials that result in poorly damped responses, it behoves one to speculate why it is often the only tuning scheme many instrument engineers are familiar with, or indeed ask if it has any concrete redeeming features at all. In fact the ZN tuning scheme, where the controller gain is experimentally determined to just bring the plant to the brink of instability is a form of model identification. All tuning schemes contain a model identification component, but the more popular ones just streamline and disguise that part better. The entire tedious procedure of trial and error is simply to establish the value of the gain that introduces

half a cycle delay when operating under feedback. This is known as the ultimate gain Ku . As it turns out, under relay feedback, most plants oscillate with a modest amplitude fortuitously at the critical frequency. The procedure is now the following:

1. Substitute a relay with amplitude d for the PID controller.

2.Kick into action, and record the plant output amplitude a and period P.

3. The ultimate period is the observed period, Pu = P, while the ultimate gain is inversely proportional to the observed amplitude,

$$Ku = 4d/\pi a$$

Having established the ultimate gain and period with a single succinct experiment, we can use the ZN tuning rules (or equivalent) to establish the PID tuning constants. Incidentally, the modified values given in Table 2.7 are improved versions of the original constants given in most textbooks which have been found to be excessively oscillatory

Specification	K_c	τ_i	τ_d
Original	$0.6K_u$	$P_u/2$	$P_u/8$
Little overshoot	$0.33K_u$	$P_u/2$	$P_u/3$
No overshoot	$0.2K_u$	$P_u/2$	$P_u/3$

Table 2.7 *Relay Tuning Chart*

Chapter 3

Experiment

Experiments require number of components, some of them are used for constructing the device, some are used to measure the parameters and some are used as source. In this experiment components are classified in three category. First category consist of heating element, fan, duct, ceramic and electronic regulator for development of the model for experiment. Second category consist anemometer and thermocouple sensor to measure the parameter used in further calculations. And the last category is the source which powers the entire system.

Heating element - Heating element is required to convert electrical energy to the heat and when air stream passes over it then heat is transferred from the heating element to the air causes to rise in temperature of air. Heating element should have higher emissivity, higher resistivity and high specific heat capacity where higher emissivity result in better transmission of energy from heating element, higher resistivity causes higher heat generation according to Joule's law of heating and high speci_c heat capacity is required to raise the temperature. Considering all these properties, Nichrome is best suited for the heating element material.
Fan - Fan is used to force the air in the duct with the desired ow rate. This forced air is responsible for conducting heat transfer from the heating element. Fan is placed in such way that direction of air must be from fan to heating element. Properties of fan which is used during experiment-

Fan size = 120mm _ 120mm _ 25mm

Fan blade outer diameter = 114mm

Fan hub diameter = 114mm

Input power = 220 V AC

Air flow rate = 100CFM

Duct - Duct is the envelope which acts as a boundary and allows air stream to flow in desired direction . Duct made of aluminum is used in the experiment because –

- Aluminum has good reactivity such that it reects maximum heat back to the air stream.
- Aluminum has low emissivity 0:04 at 20°C. This low emissivity means low absorptivity which results in low absorption as well as low emission of heat.
- Aluminum is readily available, cheap and having low weight.

Dimension of duct = 125mm _ 125mm _ 225mm

Ceramic - Since, heating element is electrically operated component of the thermoregulation device and alternating current ows through this bare heating element therefore to shield the body of the setup from the heating element electrically, Ceramic is used because it is recognized as excellent insulator.

3.1 Components

Arduino Microcontroller

Arduino is an open source, computer hardware and software company, project, and user community that designs and manufactures single-board Microcontrollers and microcontroller kits for building digital devices and interactive objects that can sense and control objects in the physical world. The project's products are distributed as open-source hardware and software, which are licensed under the GNU Lesser General Public License (LGPL) or the GNU General Public License (GPL), permitting the manufacture of Arduino boards and software distribution by anyone. Arduino boards are available commercially in preassembled form, or as do-it-yourself kits .

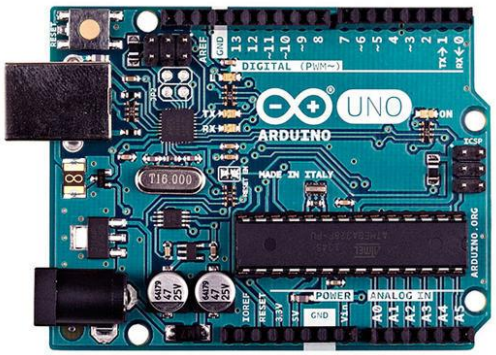


Figure 2.4 *Arduino UNO*

Digital Dimmer

The main problem with this project was to control 220 VAC current using an Arduino Uno. After searching through this problem i found that we can control the AC current using a circuit known as opto-triac circuit. In this circuit the main 2 components are Optocoupler/Optoisolator and Triac. The arduino is connected to the Optocoupler IC mainly MOC312 and the main supply of 220VAC is connected to the Triac. The Arduino Analog output Voltage controls the Optocupler which further controls the current flowing through the Triac. This is a simple circuit and has many applications. The circuit diagram is given in Figure 2.5.

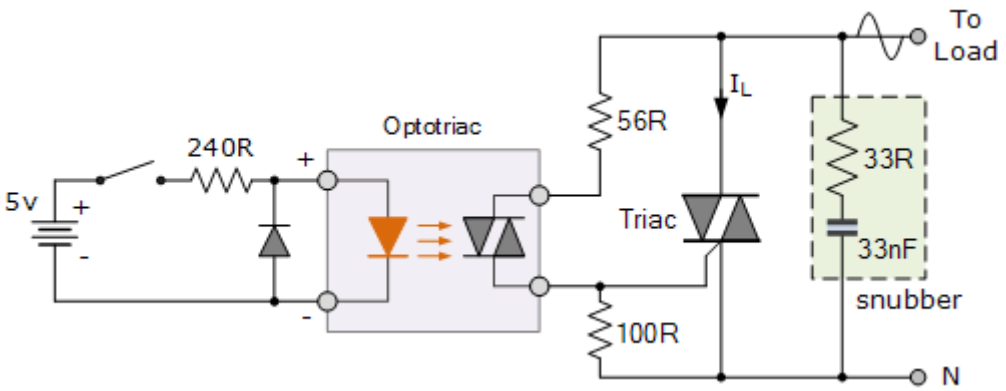


Figure 2.5 *Opto-Triac Circuit*

Instead of building this circuit i bought a module named Dgital Dimmer for this which is produced by RDL (Research Design Lab).

A dimmer switch rapidly turns a light circuit on and off to reduce the energy flowing to a light switch. The central element in this switching circuit is a triode alternating current switch, or triac. A triac is a small semiconductor device, similar to a diode or transistor. Like a transistor, a triac is made up of different layers of semiconductor material. This includes N-type material, which has many free electrons, and P-type material, which has many "holes" where free electrons can go . The triac has two terminals, which are wired into two ends of the circuit. There is always a voltage difference between the two terminals, but it changes with the fluctuation of the alternating current. That is, when current moves one way, the top terminal is positively charged while the bottom terminal is negatively charged, and when the current moves the other way the top terminal is negatively charged while the bottom terminal is positively charged.

Where DATA 0, DATA 1 , DATA 2 , DATA 3 are digital pins in the module .

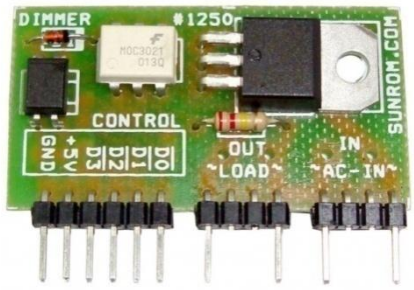


Fig 2.6 *Digital Dimmer Module*

K type Thermocouple

For sensing the temperature of the body I used K type Thermocouple. The type K is the most common type of thermocouple. It’s inexpensive, accurate, reliable, and has a wide temperature range. The type K is commonly found in nuclear applications because of its relative radiation hardness. Maximum continuous temperature is around 1,100C.

Range : -270 to 1260°C

Accuracy : +/- 2.2°C

MAX 6675

The MAX6675 performs cold-junction compensation and digitizes the signal from a type-K thermocouple. The data is output in a 12-bit resolution, SPI™-compatible, read-only format. This converter resolves temperatures to 0.25°C, allows readings as high as +1024°C, and exhibits thermocouple accuracy of 8LSBs for temperatures ranging from 0°C to +700°C.

Figure 2.7 *MAX6675 Module*

3.2 Experimental Setup

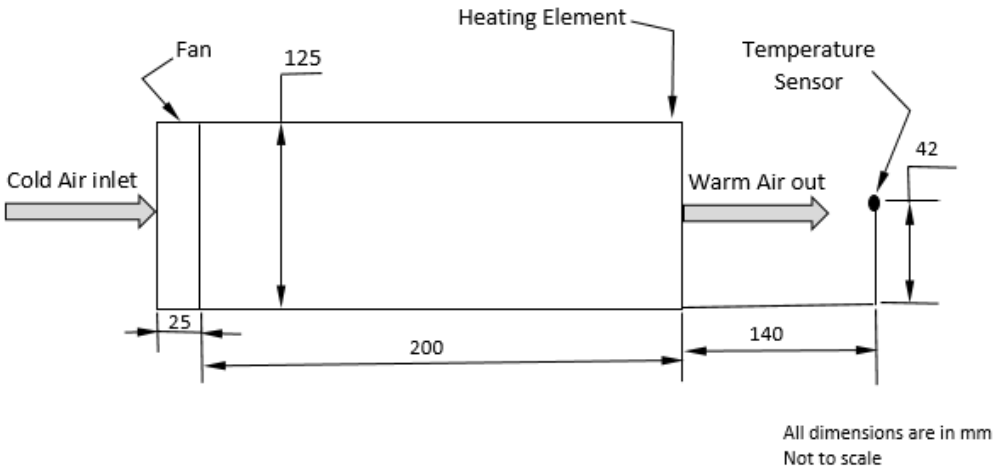


Fig 3.1 *Experimental Setup*

In the experimental setup the coil is heated by 220VAC supply and the cold air is forced into the duct using the Fan. The temperature is measured using K type

thermocouple which is fixed at a distance of 140mm from the coil. Fig3.1 shows the schematic representation of the experimental setup. The Plant function is already provided to us, which is given by

$$\Rightarrow \frac{Y(s)}{R(s)} = \frac{1.783}{s^2 + 20.39s + 1.783}$$

Figure 3.2, shows the actual setup of the experiment. Air coming from the fan flows towards the thermocouple sensor after passing through heating element where air gets heated such that air with elevated temperature pass through the sensor hence temperature higher than environmental temperature is sensed.

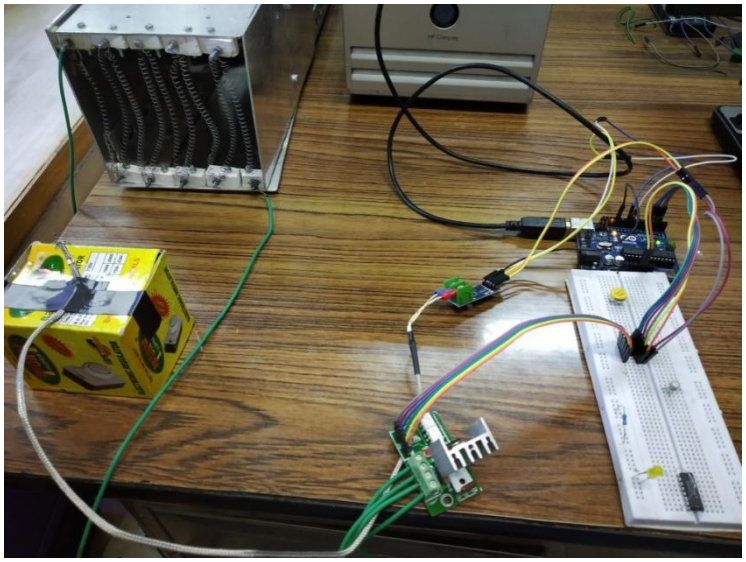


Fig 3.2 *Experimental Setup*

The Circuit Diagram for the above experimental setup is shown in Fig 3.3.

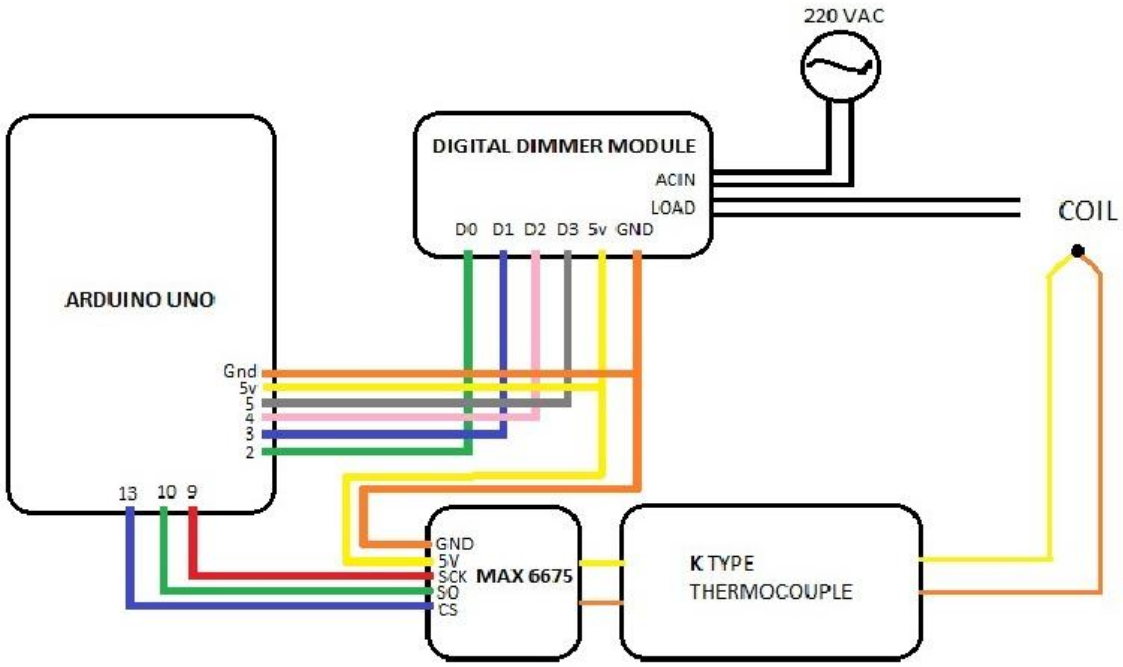


Fig 3.3 *Circuit Diagram*

3.3 Arduino Code

```
#include <SoftwareSerial.h>
#include <PID_v1.h>
#include <max6675.h>
#define CLK 9
#define DBIT 10 // so
#define CS 13
int thermo_so_pin  = 10;
int thermo_cs_pin  = 13;
int thermo_sck_pin = 9;
float timex;
MAX6675 thermocouple(9,13,10);
double Setpoint, Input, Output;
double Kp=2, Ki=0.644, Kd=1.548;
int v = 0,dimmer_input;
float Ctemp, Ftemp;
```

```
PID myPID(&Input, &Output, &Setpoint,Kp,Ki,Kd, DIRECT);
void ADC123(int dim)
{
 if(dim==0)
 {
  digitalWrite(2,1);
  digitalWrite(3,1);
  digitalWrite(4,1);
  digitalWrite(5,1);
  delay(500);
 }
  else if(dim==1)
 {
  digitalWrite(2,0);
  digitalWrite(3,1);
  digitalWrite(4,1);
  digitalWrite(5,1);
  delay(500);
 }
  else if(dim==2)
 {
  digitalWrite(2,1);
  digitalWrite(3,0);
  digitalWrite(4,1);
  digitalWrite(5,1);
  delay(500);
 }
  else if(dim==3)
 {
  digitalWrite(2,0);
  digitalWrite(3,0);
```

```
 digitalWrite(4,1);
 digitalWrite(5,1);
 delay(500);
}
 else if(dim==4)
{
 digitalWrite(2,1);
 digitalWrite(3,1);
 digitalWrite(4,0);
 digitalWrite(5,1);
 delay(500);
}
 else if(dim==5)
{
 digitalWrite(2,0);
 digitalWrite(3,1);
 digitalWrite(4,0);
 digitalWrite(5,1);
 delay(500);
}
 else if(dim==6)
{
 digitalWrite(2,1);
 digitalWrite(3,0);
 digitalWrite(4,0);
 digitalWrite(5,1);
 delay(500);
}
 else if(dim==7)
{
 digitalWrite(2,0);
```

```
 digitalWrite(3,0);
 digitalWrite(4,0);
 digitalWrite(5,1);
 delay(500);
}
 else if(dim==8)
{
 digitalWrite(2,1);
 digitalWrite(3,1);
 digitalWrite(4,1);
 digitalWrite(5,0);
 delay(500);
}
 else if(dim==9)
{
 digitalWrite(2,0);
 digitalWrite(3,1);
 digitalWrite(4,1);
 digitalWrite(5,0);
 delay(500);
}
   else if(dim==10)
{
 digitalWrite(2,1);
 digitalWrite(3,0);
 digitalWrite(4,1);
 digitalWrite(5,0);
 delay(500);
}
   else if(dim==11)
{
```

```
 digitalWrite(2,0);
 digitalWrite(3,0);
 digitalWrite(4,1);
 digitalWrite(5,0);
 delay(500);
}
   else if(dim==12)
{
 digitalWrite(2,1);
 digitalWrite(3,1);
 digitalWrite(4,0);
 digitalWrite(5,0);
 delay(500);
}
   else if(dim==13)
{
 digitalWrite(2,0);
 digitalWrite(3,1);
 digitalWrite(4,0);
 digitalWrite(5,0);
 delay(500);
}
   else if(dim==14)
{
 digitalWrite(2,1);
 digitalWrite(3,0);
 digitalWrite(4,0);
 digitalWrite(5,0);
 delay(500);
}
   else if(dim==15)
```

```
  {
    digitalWrite(2,0);
    digitalWrite(3,0);
    digitalWrite(4,0);
    digitalWrite(5,0);
    delay(500);
  }
  else
  {
    Serial.print("Error");
  }
}

void setup()
{
  Input = analogRead(0);
  Setpoint = 43;
  Serial.begin(9600);
  pinMode(CLK, OUTPUT);
  pinMode(DBIT, INPUT);
  pinMode(CS, OUTPUT);
  pinMode(2,OUTPUT);
  pinMode(3,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  digitalWrite(CS, HIGH);
  digitalWrite(CLK, LOW);
  myPID.SetMode(AUTOMATIC);
  Serial.println("CLEARDATA");
  Serial.println("LABEL,Real Time,Time,Temperature,SetPoint");
  Serial.println("RESETTIMER");
```

```
 // pinMode(thermo_vcc_pin, OUTPUT);
 // pinMode(thermo_gnd_pin, OUTPUT);
}

void loop()
{
  Ctemp =  v * 0.25;
  Input = thermocouple.readCelsius();
  myPID.Compute();
  dimmer_input = Output/16;
  ADC123(dimmer_input);
  timex = millis()*0.001;
  Serial.print("DATA,TIME,");
  Serial.print(timex);
  Serial.print(",");
  Serial.print(thermocouple.readCelsius());
  Serial.print(",");
  Serial.println(Setpoint);
  }

int spiRead()
{
  int value = 0;
  digitalWrite(CS,LOW);
  delay(2);
  digitalWrite(CS,HIGH);
  delay(220);
  digitalWrite(CS,LOW);
  digitalWrite(CLK,HIGH);
  delay(1);
  digitalWrite(CLK,LOW);
```

```
  for (int i=14; i>=0; i--) {
    digitalWrite(CLK,HIGH);
    value += digitalRead(DBIT) << i;
    digitalWrite(CLK,LOW);
  }
  if ((value & 0x04) == 0x04) return -1;
return value >> 3;
}
```

3.4 Evaluation of PID Coefficients

1. Using MATLAB PID Tuner

The Step response of the Plant function is plotted in MATLAB. As you can see the plant's unit step response looks like a step response of a First Order System. But as we know that the plant is a Second Order System.

The reason that the step response of Plant Function is coming out to be like the step response of a First Order is because the poles of the TF are coming out at -0.049 and -11.38. This means that the pole which is closer to the origin has least significance and hence the TF acts as a FOPDT system.

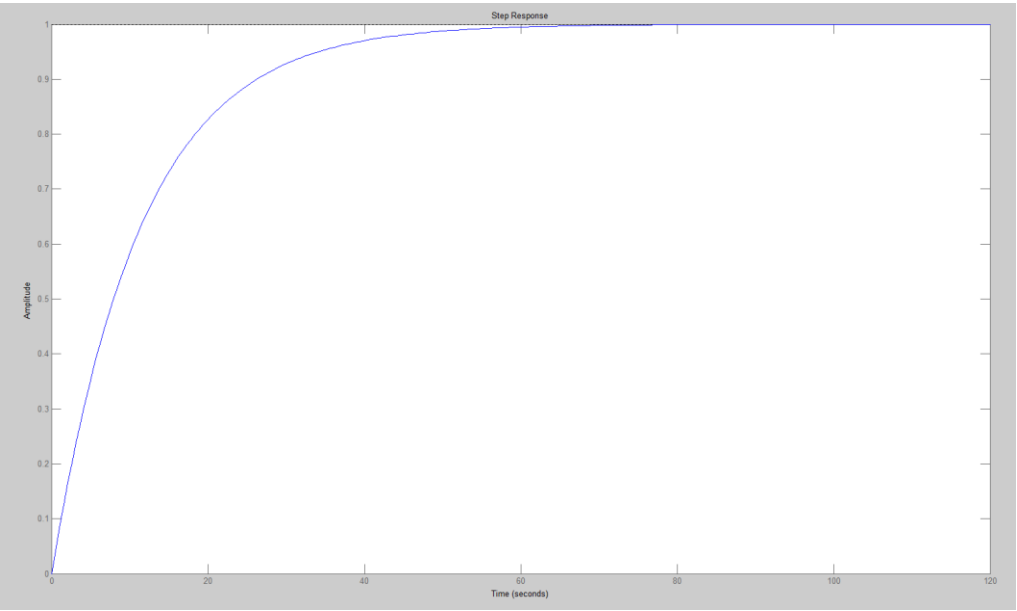


Fig 3.3 *Step Response of Plant*

MATLAB Code to generate the step response is given by,

```
clc
clear all
sys = tf([1:783],[1 20:39 1:783]);
step(sys)
```

Now the PID coefficients are calculated using the toolbox in MATLAB known as PID Tuner.

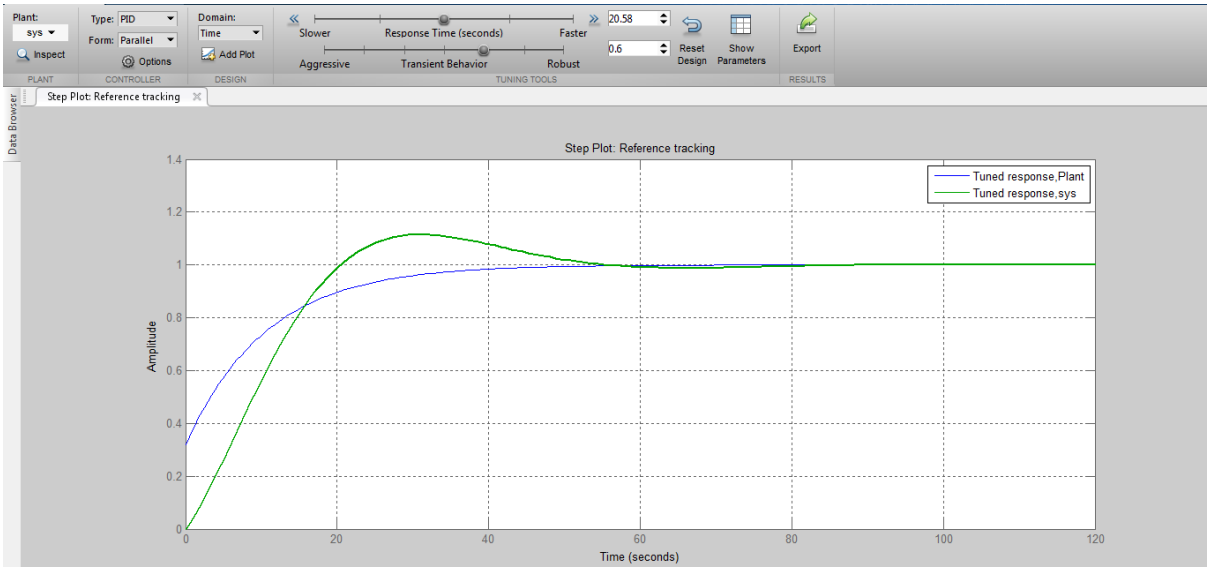


Fig 3.4 *PID Tuner Toolbox*

Fig 3.5 *PID Tuner Toolbox Results*

The values of PID Parameters are as follows:-

Kp = 0.06636

Ki = 0.058883

Kd = 0

Now these values are put in the PID Controller.

2.Using Skogestad's Tuning Formula

For Skogestad's Method of PID Tuning we are using Dead time. Now to calculate the dead time we can do it theoretically or practically. The problem with calculating the dead time practically is that sampling period for the temperature should be very low. This much low sampling period is not achievable by the Arduino's analog input pins. Hence , we calculate the dead time theoretically.

From Figure we can see that the Thermocouple is placed at a distance of 140mm from the coil and the rate at which air is flowing through the coil is $4.26ms^{-1}$. Hence we can calculate the time which would be taken by the heated air to reach the thermocouple.

The dead time is calculated as follows,

$140*10^{-3}/4.26 = 0.03286$ seconds

$T_d = 0.03286$

$T_1 = 11.38$

$T_2 = 0.049$

After various calculation using the above three values and putting them in Table 2.6 we get

Kp = 1, Ki = 3.46, Kd = 0.0406

3.Using Ziegler Open Loop Tuning Method

In ZN Open Loop Tuning Method the Kp,Ki,Kd are calculated using the Table 2.3.

The value of Kp = 1, Ki = 1.5625, Kd = 0.016.

Now these parameters are inserted in the controller and the results are plotted.

4.Relay Tuning

The procedure is now the following:

1. Substitute a relay with amplitude d for the PID controller.

2.Kick into action, and record the plant output amplitude a and period P.

3. The ultimate period is the observed period, Pu = P, while the ultimate gain is inversely proportional to the observed amplitude,

$$Ku = 4d/\pi a$$

Now calculate the Ultimate Gain and Period from Figure 3.5 and Table 2.7

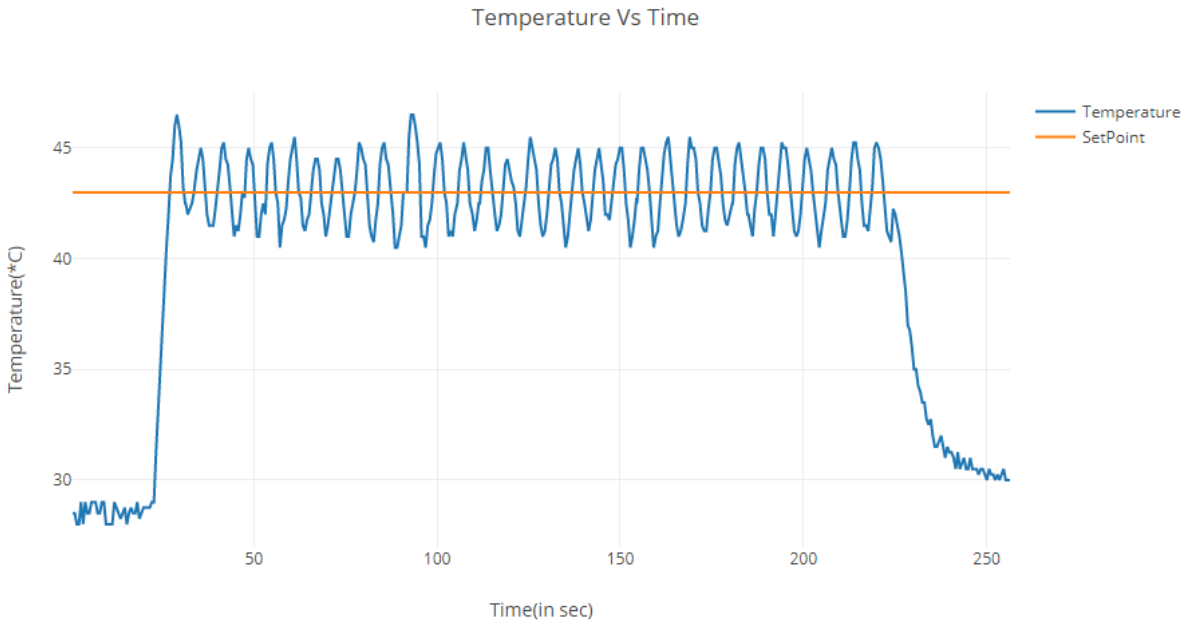


Figure 3.5 *Replacing PID controller with Relay*

The values of PID parameters comes out to be

Kp = 2, Ki = 0.644, Kd = 1.548

Now these parameters are inserted in the controller and the results are plotted.

5.Cohen Coon Method

For the Cohen Coon Method first we have to plot the open loop response of the Plant.

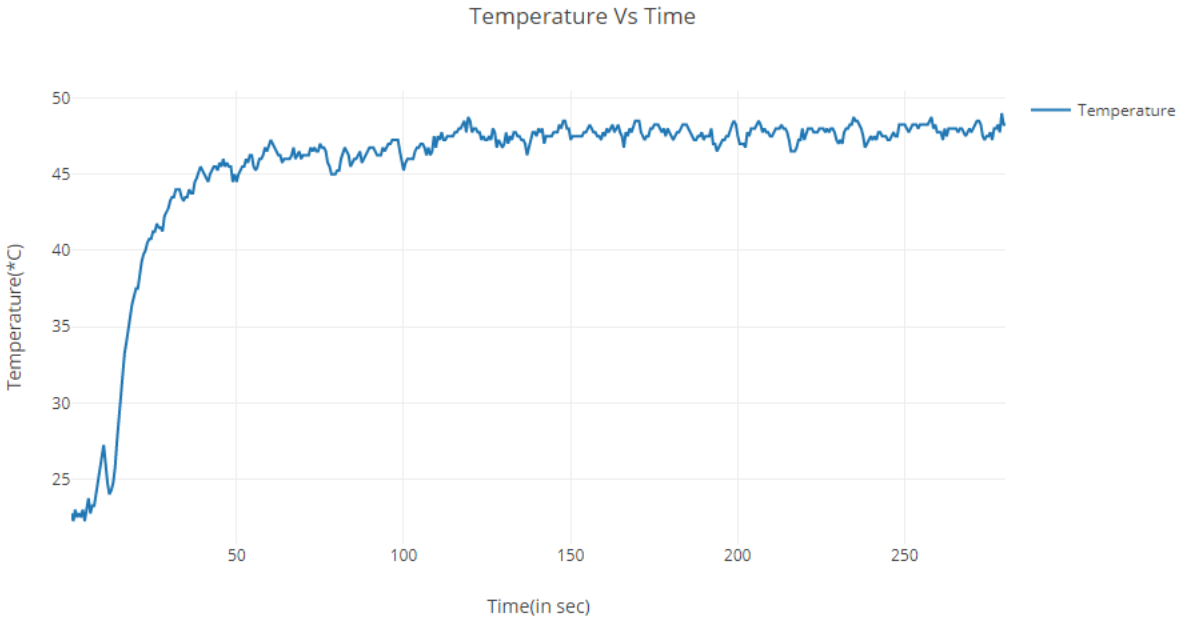


Fig 3.6 *Open Loop Response of the Plant*

Now using the Cohen Coon formulas we calculate the value of PID Parameters.

The parameters comes out to be

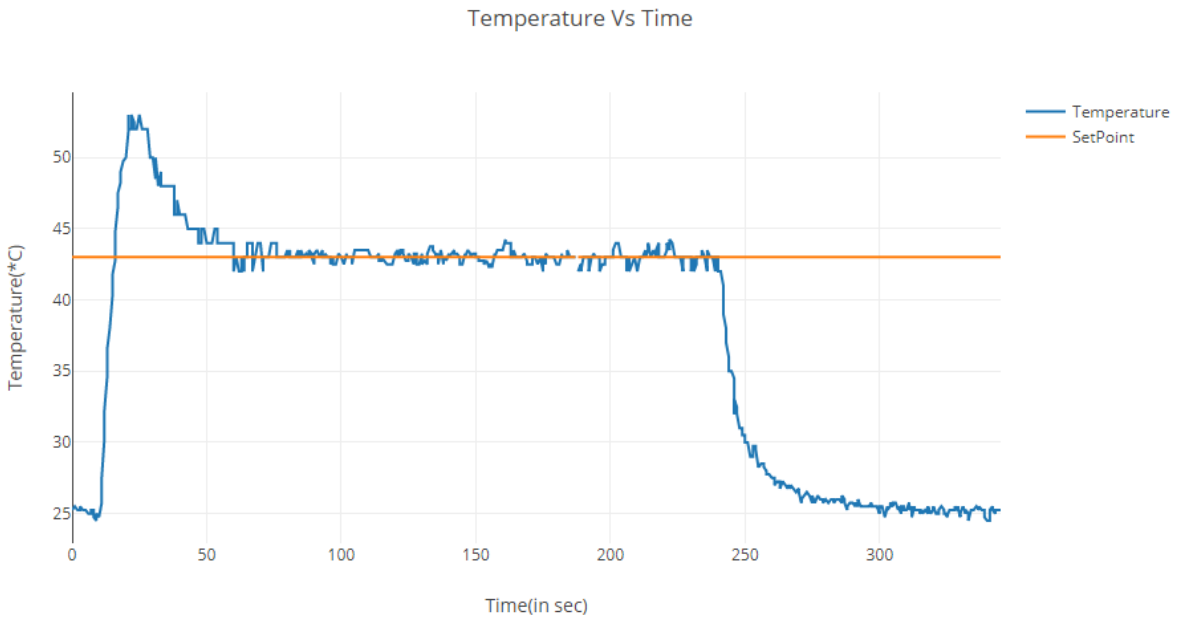
Kp = 0.25, Ki=3.1805, Kd=0.0029075

Now these parameters are inserted in the controller and the results are plotted.

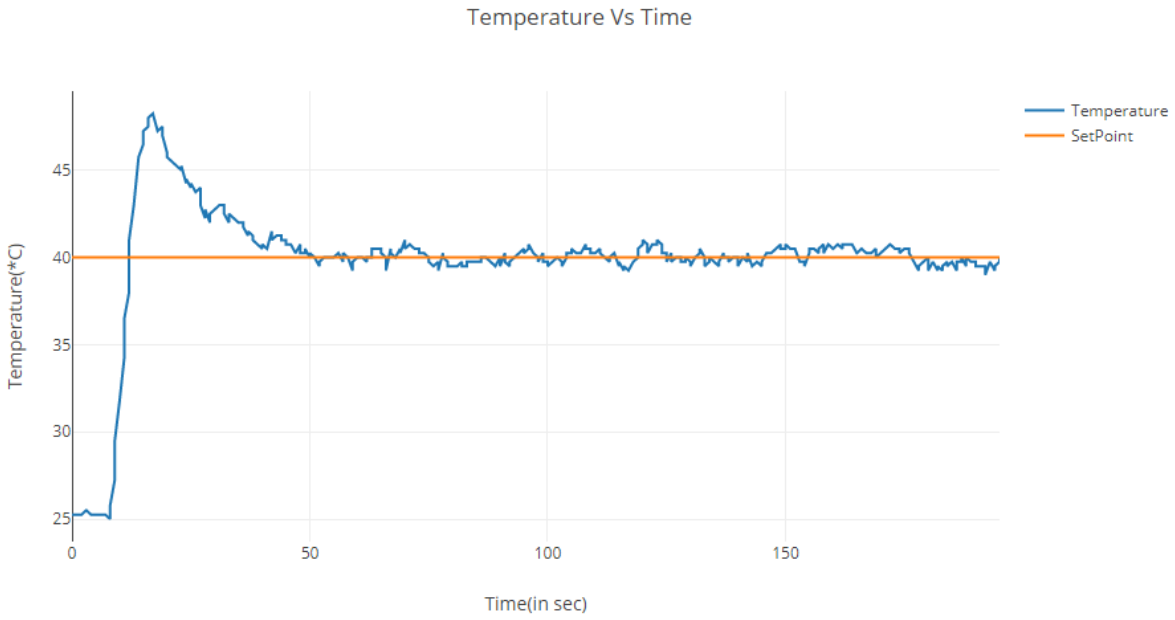
Chapter 4

Results and Discussion

1.Matlab PID Tuner Toolbox

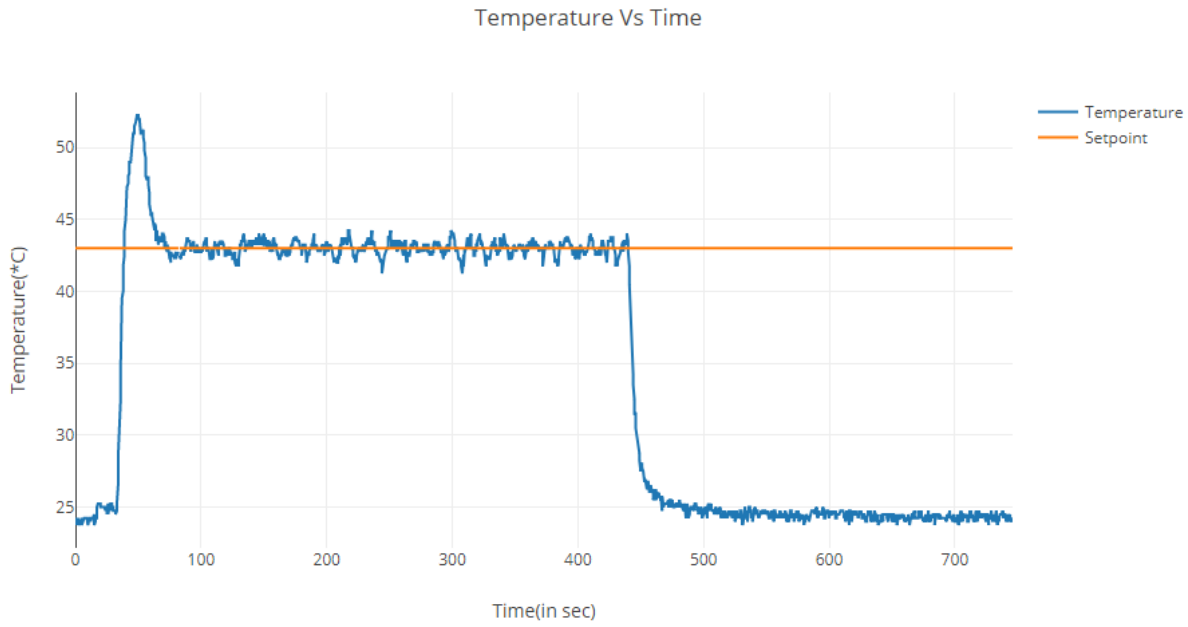


Setpoint Value	43°Celsius
Rise Time	16 seconds
% Overshoot	20.9%
Settling Time	50 seconds
Total Running Time	345 seconds
Room Temperature	25.25°Celsius
Power On Time	11 seconds
Power Off Time	240 seconds
Kp	5.00
Ki	5.63
Kd	0.00

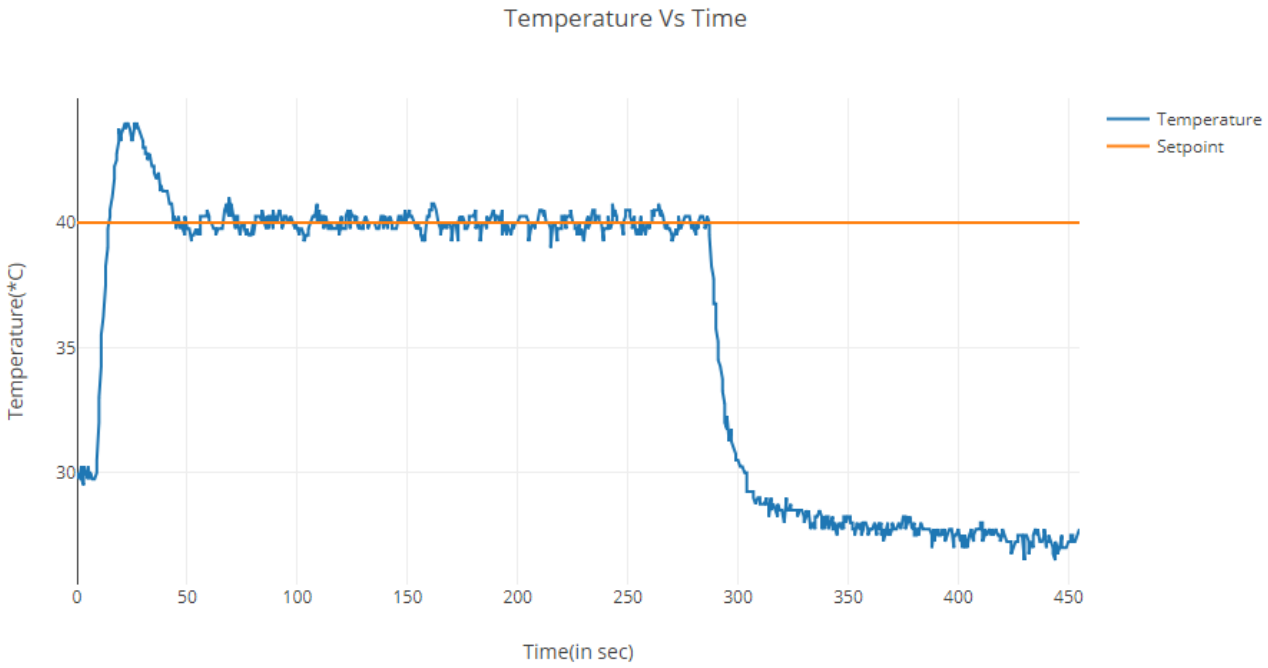


Setpoint Value	40°Celsius
Rise Time	12.5 seconds
% Overshoot	20%
Settling Time	33 seconds
Total Running Time	195 seconds
Room Temperature	25.25°Celsius
Power On Time	8 seconds
Power Off Time	195 seconds
Kp	5.00
Ki	5.63
Kd	0.00

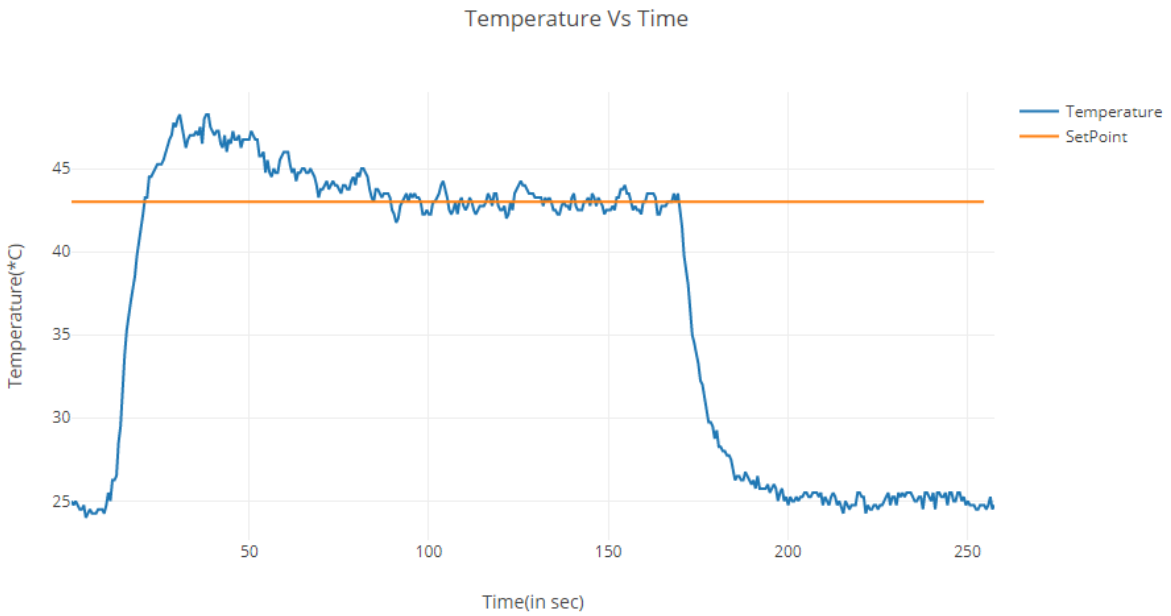
2 .Skogestad's Tuning Method



Setpoint Value	43°Celsius
Rise Time	6 seconds
% Overshoot	20.93%
Settling Time	47 seconds
Total Running Time	746 seconds
Room Temperature	25°Celsius
Power On Time	33 seconds
Power Off Time	441 seconds
Kp	2.00
Ki	6.92
Kd	0.082

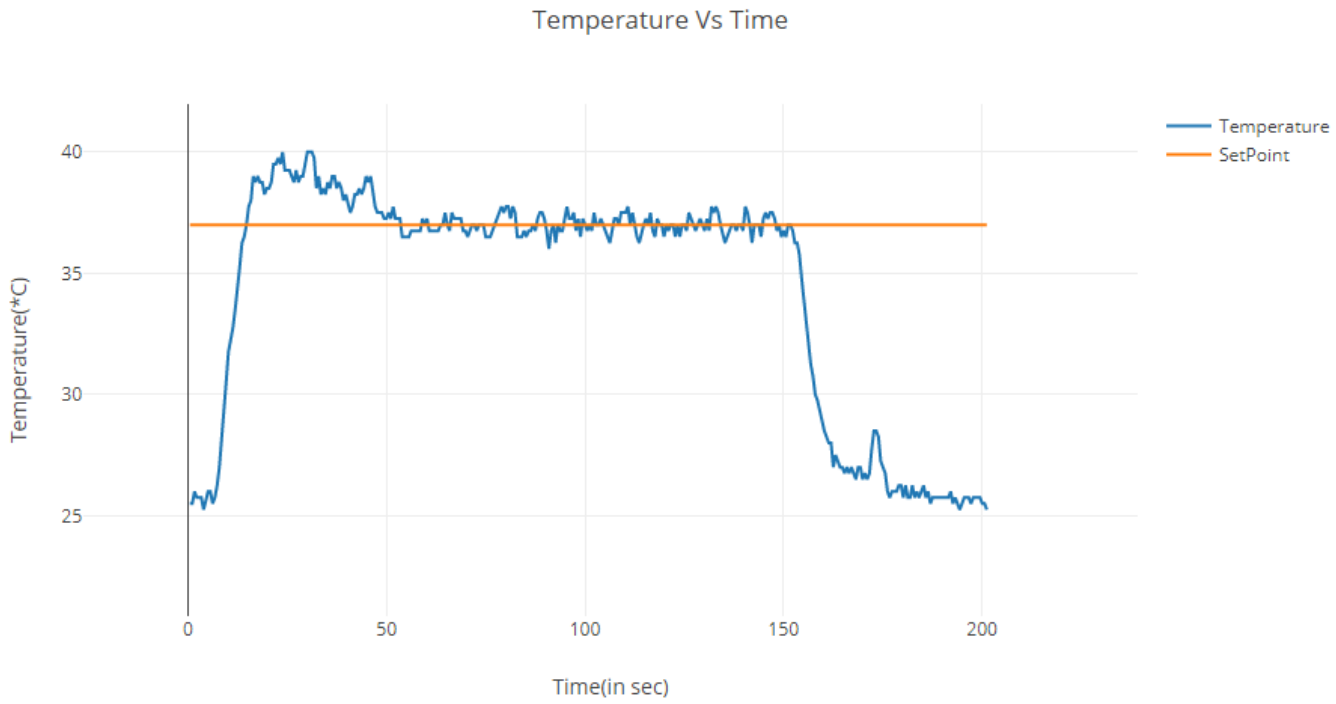


Setpoint Value	40°Celsius
Rise Time	6 seconds
% Overshoot	10%
Settling Time	34 seconds
Total Running Time	455 seconds
Room Temperature	30°Celsius
Power on Time	10 seconds
Power off Time	287 seconds
Kp	1.00
Ki	3.46
Kd	0.0406



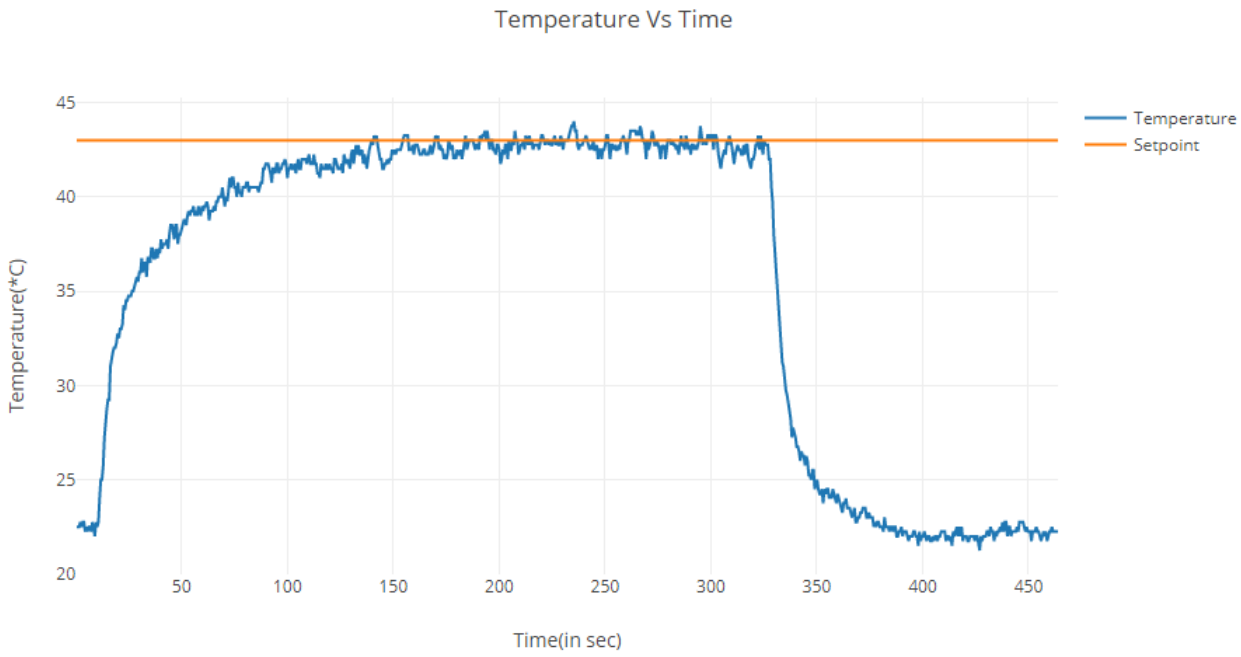
Setpoint Value	43°Celsius
Rise Time	15 seconds
% Overshoot	9.30%
Settling Time	73.91 seconds
Total Running Time	257.52 seconds
Room Temperature	25°Celsius
Power on Time	10.20 seconds
Power off Time	169.39 seconds
Kp	1.00
Ki	3.46
Kd	0.0406

-



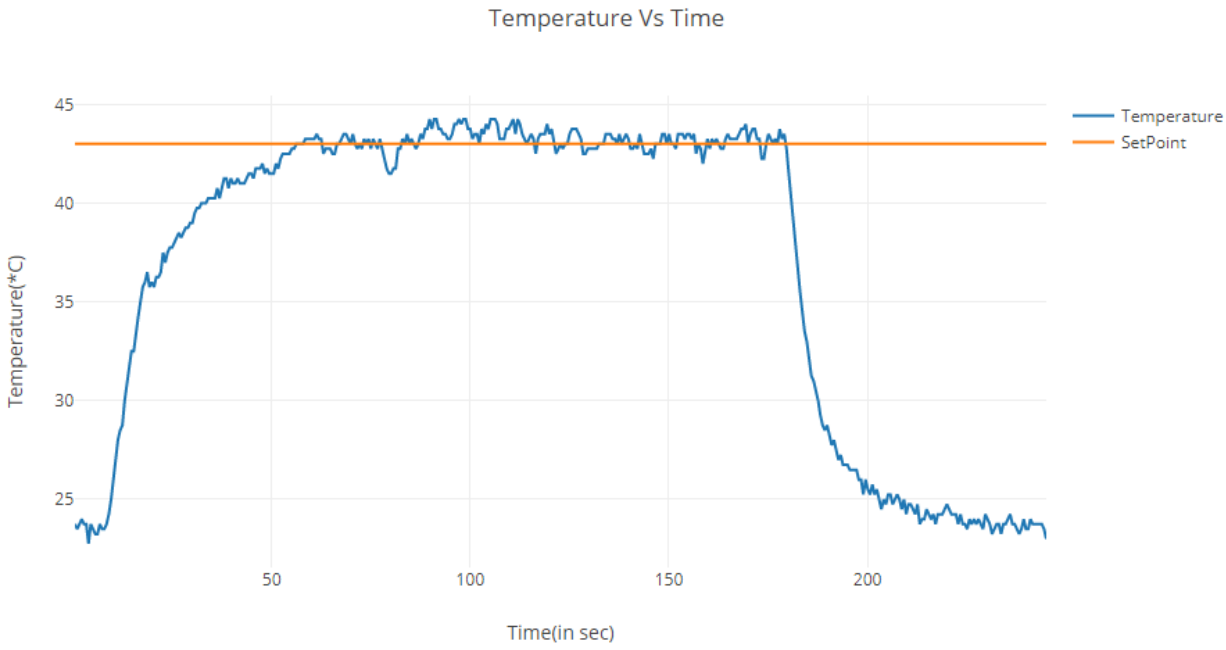
Setpoint Value	37°Celsius
Rise Time	7.39 seconds
% Overshoot	8.1%
Settling Time	42.64 seconds
Total Running Time	200.10 seconds
Room Temperature	26°Celsius
Power on Time	7.36 seconds
Power off Time	152.34 seconds
Kp	1.00
Ki	3.46
Kd	0.0406

3. Relay Tuning Method



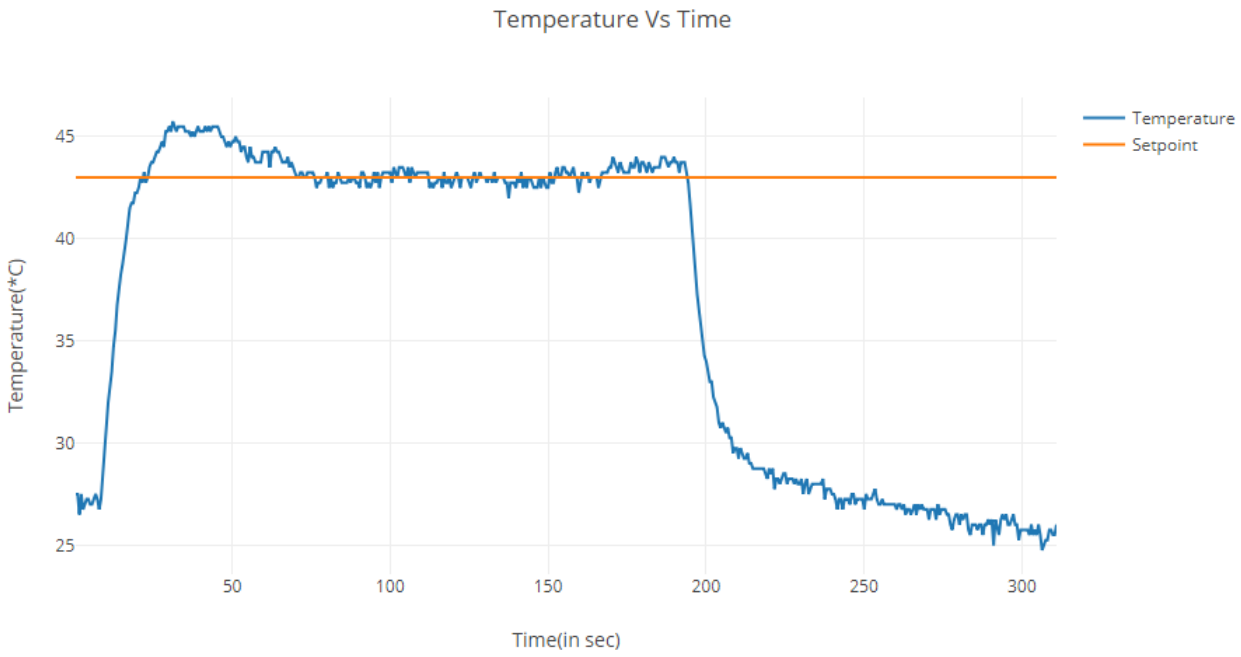
Setpoint Value	43°Celsius
Rise Time	129.06 seconds
% Overshoot	1.744%
Settling Time	143.27 seconds
Total Running Time	463.92 seconds
Room Temperature	22.5°Celsius
Power on Time	10.77 seconds
Power off Time	326.89 seconds
Kp	2.00
Ki	0.644
Kd	1.548

4. Ziegler Nichols Open Loop Tuning Method



Setpoint Value	43°Celsius
Rise Time	45.48 seconds
% Overshoot	1.744%
Settling Time	46.62 seconds
Total Running Time	245.01 seconds
Room Temperature	24°Celsius
Power on Time	9.06 seconds
Power off Time	179.63 seconds
Kp	1.00
Ki	1.5625
Kd	0.016

5. Cohen Coon Tuning Method



Setpoint Value	43°Celsius
Rise Time	12.51 seconds
% Overshoot	5.23%
Settling Time	59.70 seconds
Total Running Time	310.97 seconds
Room Temperature	27.5°Celsius
Power on Time	8.49 seconds
Power off Time	194.41 seconds
Kp	0.25
Ki	3.1805
Kd	0.0029075

Chapter 5

Conclusion

From the above simulations we conclude that depending on the need of the user we can use different parameters for the controller.

Depending on the case we can assign PID parameters to them accordingly.

CASE I : In this case there is a need of less settling time and the overshoot percentage is of less importance Skogestad's tuning parameters are preferred. Less Settling Time , Less Rise Time, High Overshoot.

Settling Time = 47 seconds

Overshoot percentage = 20.93%

Skogestad's tuning parameters :-

Kp = 2, Ki = 6.92, Kd = 0.082

CASE II : In this case there is a need of less overshoot percentage and the settling time is of less importance Relay Tuning or parameters are preferred. Less Overshoot , Large Rise Time, Large Settling Time.

Settling Time = 143.27 seconds

Overshoot percentage = 1.44 %

Relay Tuning parameters :-

Kp = 2, Ki = 0.644, Kd = 1.548

CASE III : In this case a balance between the settling time and the overshoot is found. For this case Ziegler Nichols Open loop Tuning parameters are preferred as they have not so large overshoot and the settling time is also not that small.

Settling Time = 46.62 seconds

Overshoot percentage = 1.44%

Ziegler Nichols Open loop Tuning parameters :-

Kp = 1, Ki = 1.5625, Kd = 0.016

Chapter 6

Bibliography

Fleming, P. J. (2001). Genetic algorithms in control systems engineering. *Department of Automatic Control and Systems Engineering* .

Hang, C. a. (1991). A comparative performance study of PID auto-tuners. *IEEE Control Systems* , 41--47.

Hang, C. C. (1991). Refinements of the Ziegler--Nichols tuning formula. *IET* , 111--118.

Leva, A. (1993). PID autotuning algorithm based on relay feedback. *IET* , 328--338.

Shahrokhi, M. a. (2013). Comparison of PID controller tuning methods. *Department of Chemical \& Petroleum Engineering Sharif University of Technology* , 1--2.

CSIR- CSIO

CSIR- Central Scientific Instruments Organisation

Sector 30-C, Chandigarh-160030 India

Training Diary

Name AREKH TIWARI
College/Department N.I.T JALANDHAR
University NIT JALANDHAR
Training Period 11.6.2018 to 28.7.18
CSIO Id A2297.
CSIO Guide Sanjeev Soni
Division V2

Contact Information:
Blood Group O+ve
Mobile number 7725908289
Emergency contact name and number VIJAY TIWARI (9814162283)
Email id arokntiwari@gmail.com
Residential phone number 9814162283
Residential address 9A-D, R.C.F, KAPURTHALA, PUNJAB

Working days: Monday to Friday
Working hours: 9.00am to 5.30pm

11.6.18

CSIR-CSIO Weekly Training Diary: Week 1 starting

INTRO :-

The aim of the work concentrates around the tuning of a thermoregulation device. However, this project carries small part in the development of the device. The work shows the variation of response characteristics due to change in PID parameters K_p, K_i, K_d and proposed the scheme to implement the tuned PID parameters to control over the temperature during entire operation and to raise the patients body temperature to the normal level during the entire perioperative period. This device is important during the peri-operative period as during surgery, anaesthetic drugs are given to the patient to make them insensitive of surgical procedure but these also affect the thermoregulatory system of the body and result in fall in temperature.

Signature of Guide

Comparison of various PID tuning techniques

1.) Manual PID tuning

Primitive way of tuning the PID parameters. Often results in the system going drastically unstable while performing the PID tuning procedure

2.) ZN tuning Method (Closed loop)

The ZN rule is a heuristic PID tuning rule that attempts to produce good values for the three PID gain parameters. Given 2 measured feedback loop parameters derived from measurements:

1. T_u : Ultimate Period
2. K_u : Ultimate Gain

The values of PID parameters can be calculated

3.) ZN open loop tuning

This method is based on the open loop response of the system. This open loop response gives us T_d & T_c. Which are used to calculate PID parameters.

Signature of Guide

Comparison of various PID tuning techniques

4.) Cohen Coon Method

In this method the process reaction curve is obtained first, by an open loop test
The process dynamics is appx. as a FOPDT

$T_m = \frac{3}{2}(t_2 - t_1)$

$d_m = T_2 - T_m$

5.) Using genetic algorithm

In this method we define a fitness function which is usually the performance index, IAE, ISE, ITAE or ITSE.
Then we use GA to optimise these performance indexes.

6.) Skogestad's Tuning algorithm

The TF is defined as $T(s) = \frac{1}{T_c s + 1} e^{-\tau s}$

Using the tuning chart we calculate PID parameters

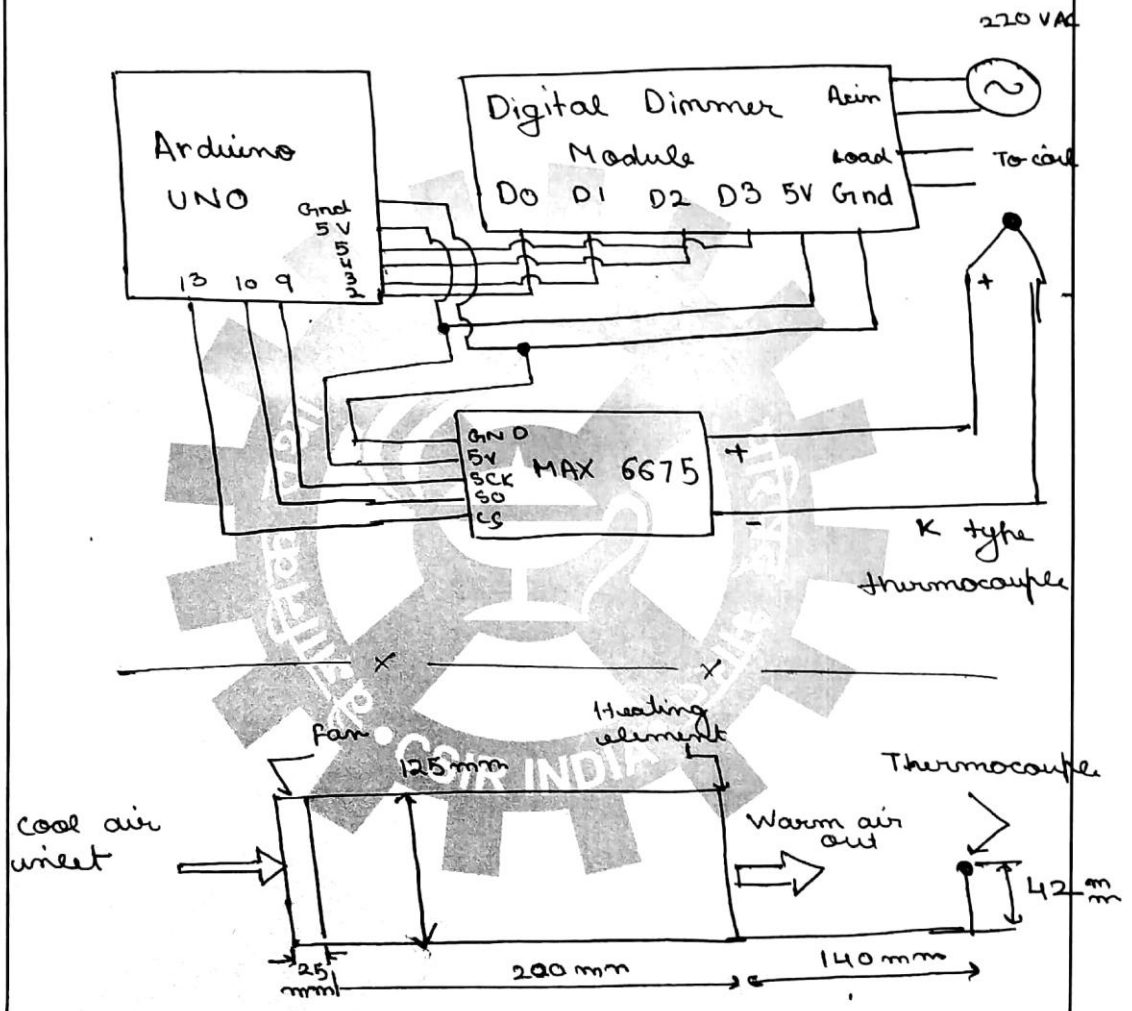
7.) Relay Tuning

In this method a relay is placed at the place of PID controller and T_u & K_u is calculated from that.

Signature of Guide

CSIR-CSIO Weekly Training Diary: Week 4 starting Monday,

CIRCUIT DIAGRAM / BLOCK DIAGRAM



Experimental Setup

Signature of Guide

CSIR-CSIO Weekly Training Diary: Week 5 starting Monday,

INTRO

This new project is based on development of a VI (Virtual instrument) in LabVIEW which display the current, ~~volts~~ through the lamp, Voltage across the lamp, Temperature from the thermocouple. This should all be done using NI DAQ 6211 and LABVIEW. This project should be able to synchronise the measurements i.e. current, voltage, temperature and also the switching of the lamp. For these purposes we will design PCB circuits using relays, transistor etc.

Signature of Guide

CURRENT SENSING

The current through the ~~coil~~ wire would be sensed using a hall sensor. The hall sensor will generate voltage ∝ to the current flowing through the wires

VOLTAGE SENSING

The voltage across the lamp would be sensed using a Voltage divider CKt. The Resistance which would be applied across the lamp would be greater than lamps Resistance to avoid current loss.

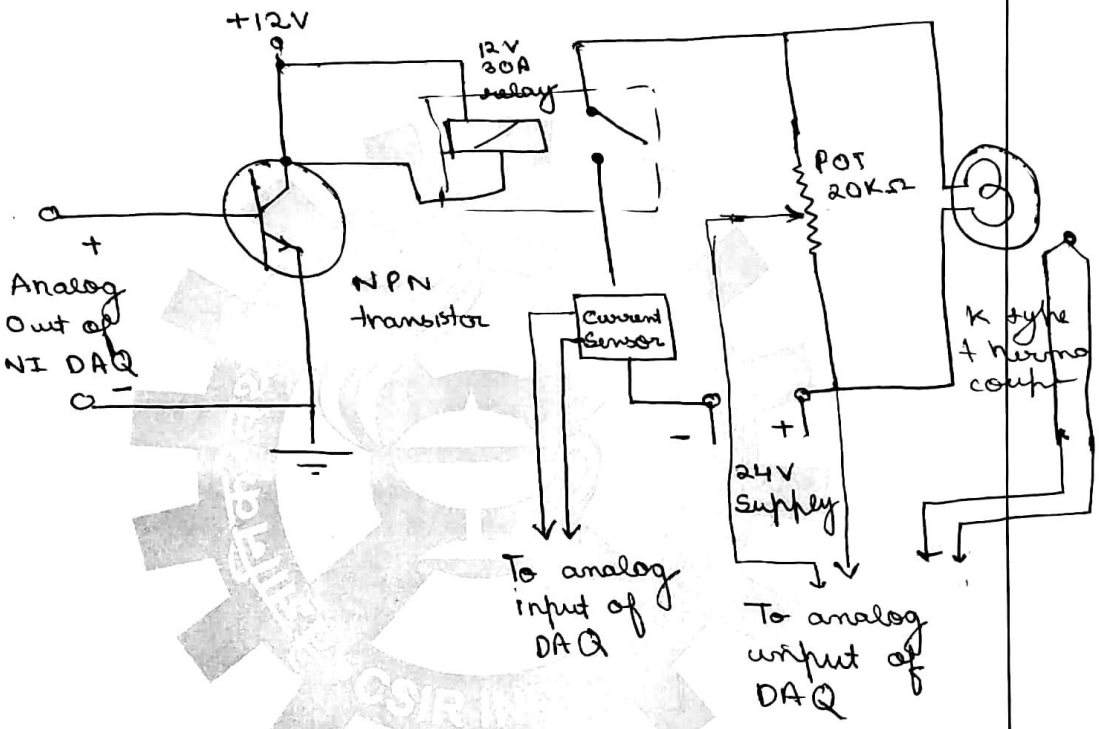
RELAY SWITCHING

As the DAQ couldnot provide enough current to switch a 30A relay. We would be using a Transistor + Relay CKt. with an external power supply of 12Volts.

Signature of Guide

CSIR-CSIO Weekly Training Diary: Week 7 starting Monday,

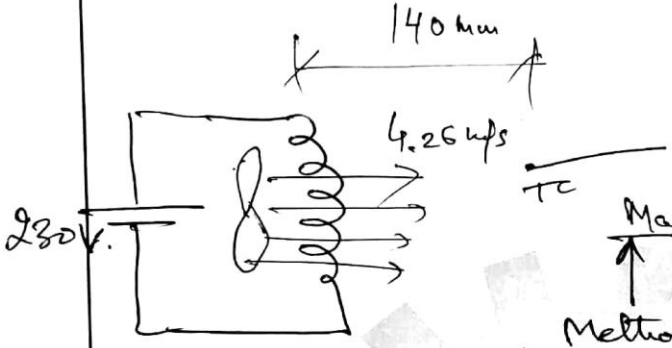
CIRCUIT / BLOCK DIAGRAM



This above ckt. diagram shows the assembling of all the three factors

- Current Sensing
- Voltage Sensing
- Temp Sensing
- Synchronus Switching

Signature of Guide



Method 1 → $K_P K_I K_D$

'T' Control 2 →

3 →

Evaluation of K_P, K_I, K_D

(a) Dead time = $\frac{\text{Distance}}{\text{Velocity of Air}} = \frac{0.140\text{ m}}{4.26} = 0.032\text{ S}$

$= 32\text{ mS}$

(b) TF → open loop step response → time constant

(c) Tuning chart of each method ⇒ $K_P K_I K_D$

(d) Arduino — pgm I/P → Set point

$K_P K_I K_D$

Ziegler Nicolas – open loop tuning

1.74% of Tset. ⇒ 0.5 to 1°C Control

Signature of Guide

CSIR-CSIO Weekly Training Diary: Week 9 starting Monday,

Heater Power

230 V, 4.6 A.

① longer duration op^{n}

② Power consumption Plot.

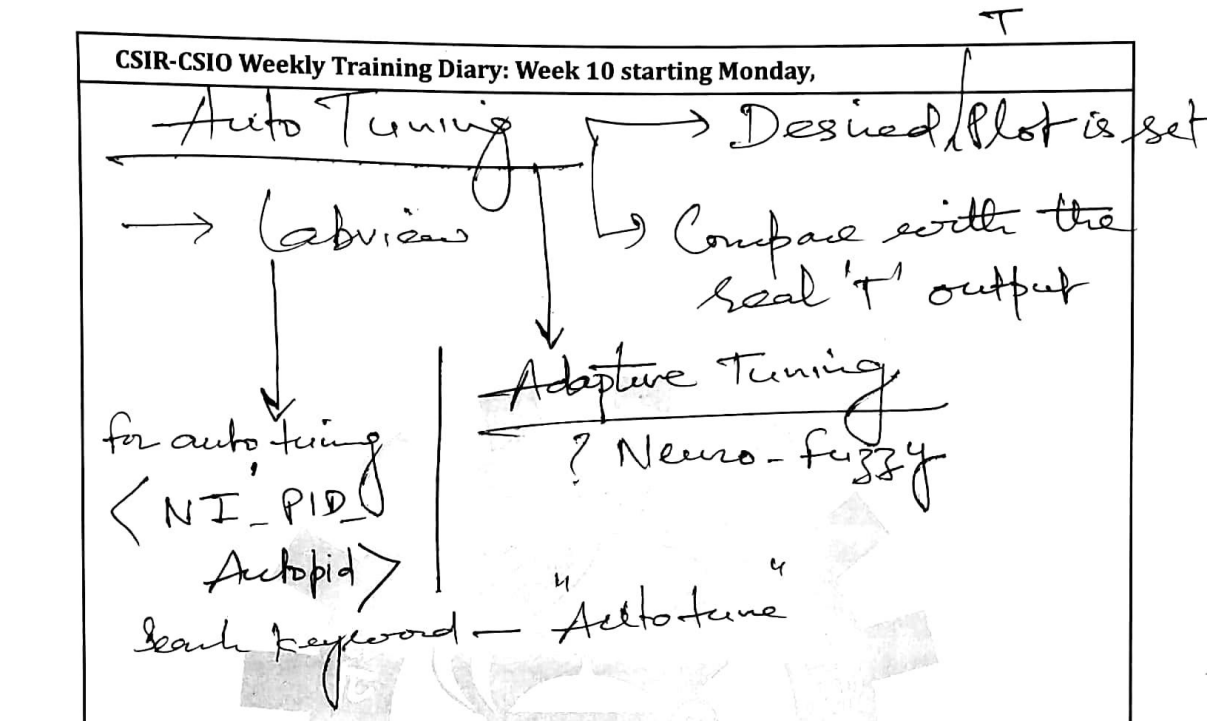
To log the temp data, use

PLX-DAQ (installed)

(serial Print)

Signature of Guide

CSIR-CSIO Weekly Training Diary: Week 10 starting Monday,



VI front End

Signature of Guide