POLITECNICO
MILANO 1863

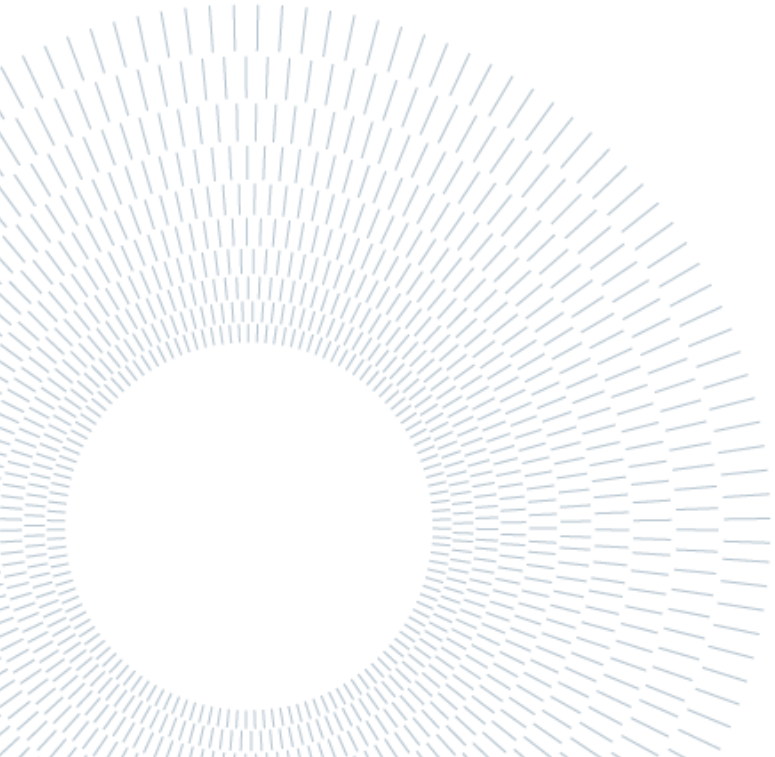
SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

DEVELOPMENT OF TPI TEMPERATURE CONTROL ALGORITHM FOR THERMOSTAT FOR MAXIMUM COMFORT & HIGH ENERGY EFFICENCY

TESI DI LAUREA MAGISTRALE IN
ENERGY ENGINEERING
INGEGNERIA ENERGETICA

Author: **Saqib Ali** **(10628468)**

Student ID: 943780
Advisor: Antonino Di Gerlando (Politecnico di Milano)
Co-advisor: Roberto La Capruccia(Fantini Cosmi)
Academic Year: 2021-22



Abstract

Nowadays, the world is not only moving away from conventional fossil fuels and shifting towards renewable sources, but it is also getting more concerned and conservative in energy saving. Moreover, while saving energy, the desire is to have a healthier lifestyle with better comfort.

This project is all about saving energy and providing better comfort to users. A thermostat is a device that controls the temperature and conventional ON/OFF thermostats are long been in use but now with the changing world, they should be replaced with new and more efficient thermostats with highly precise temperature control.

It's the project of developing a new state-of-the-art temperature control algorithm that can be integrated into thermostats with the benefits of 10 % saving energy and better comfort compared to conventional thermostats. With this aim in mind MATLAB, Simulink and DesignBuilder are used as simulation environments and developed the model uses different blocks representing our system.

Furthermore, the project includes a full study of the time proportional integral (TPI) temperature control system which should be compatible with all energy classes and as far as heating equipment is concerned it is designed to be compatible with radiator and fan coil. The temperature controller designed is feedback type and has auto-tuning capabilities and it is inspired by Relay-Based Tuning and Ziegler-Nichols' Process Reaction Curve auto-tuning method.

Key-words: Time proportional integral (TPI) control, Temperature control algorithm TPI controlled thermostat, Temperature control system

Abstract in italiano

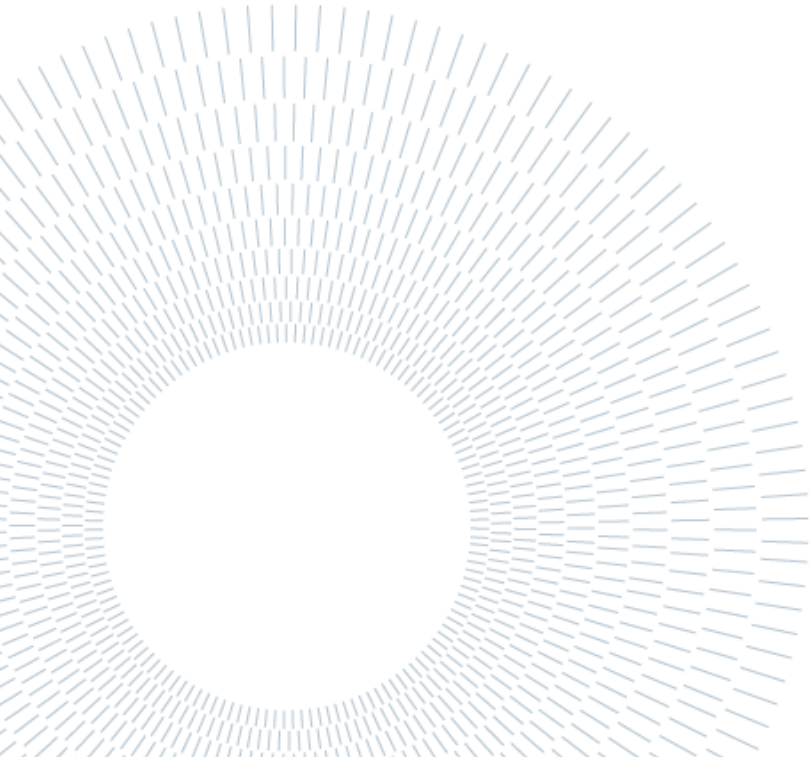
Al giorno d'oggi, il mondo non solo si sta allontanando dai combustibili fossili convenzionali e si sta spostando verso le fonti rinnovabili, ma è anche più attento e prudente nel risparmio energetico. Inoltre, pur risparmiando energia, il desiderio è quello di avere uno stile di vita più sano con un miglior comfort.

Questo progetto è incentrato sul risparmio energetico e sul miglioramento del comfort degli utenti. Un termostato è un dispositivo che controlla la temperatura e i tradizionali termostati ON/OFF sono in uso da tempo, ma ora con il mondo che cambia, dovrebbero essere sostituiti con termostati nuovi e più efficienti con un controllo della temperatura altamente preciso.

È il progetto di sviluppare un nuovo algoritmo di controllo della temperatura all'avanguardia che possa essere integrato nei termostati con i vantaggi del 10% di risparmio energetico e un migliore comfort rispetto ai termostati convenzionali. Con questo obiettivo in mente MATLAB, Simulink e DesignBuilder sono utilizzati come ambienti di simulazione e sviluppato il modello utilizza diversi blocchi che rappresentano il nostro sistema.

Inoltre, il progetto prevede uno studio completo del sistema di termoregolazione TPI (Time Proportional Integrale) compatibile con tutte le classi energetiche e per quanto riguarda le apparecchiature di riscaldamento è progettato per essere compatibile con radiatori e ventilconvettori. Il controller di temperatura progettato è di tipo feedback e dispone di funzionalità di sintonizzazione automatica ed è ispirato al metodo di sintonizzazione basato su relè e alla curva di reazione del processo di Ziegler-Nichols.

Key-words: Controllo integrale proporzionale del tempo (TPI), Algoritmo di controllo della temperatura Termostato controllato da TPI, Sistema di controllo della temperatura



Contents`

Introduction

A thermostat is a device that is used to control the temperature of the room or building. It's been long in use since the 1930s and now's a day the market for the better thermostat is rising considering the rising fuel price and desire for a high level of comfort. Solely taking into account two factors better comfort and high energy efficiency, this project's goal is set to achieve those goals.

Nowadays, as the world is moving away from conventional fossil fuels and moving to renewable energy sources, that is inspiring and in the meanwhile dire need for humankind so to tackle climate change, which is approaching us quickly. But the process of shifting from fossil fuels to renewables is not that quick in order to meet the energy demand of the world. Therefore, there is another approach to using energy in an efficient way so that we could have enough time for the shift of dependency from fossil fuels to green energy.

This gives birth to the idea of smart and energy-efficient buildings, if we consume energy in an efficient way, we don't need to have to burn more fossil fuels too much energy to meet the demand, hence fewer carbon emissions. One of the ways to make building more energy efficient is to have a better controlling device to control the temperature of the buildings, therefore better temperature control algorithm can play a significant part in saving energy and saving the world as well.

In the European Union, the most share of energy consumption is for space heating which is around 62 % [1] which is a lot compared to other consumption like water heating, cooking, or cooling. In the light of above discussion space heating or temperature control of ambient is important for energy saving. Therefore, this project's main aim is to control the temperature so that it does not fluctuate much, which also means not wasting energy. If a user's set point is 20 °C, and the temperature swings between 22 °C & 18 °C, which each degree increases in temperature from 20 °C, it is undesirable and you might not feel the difference, but you might feel it in your gas or electricity bills. The development of the time proportional integral (TPI) temperature control algorithm which is the aim of the project known to save 10 % more energy than the conventional ON/OFF thermostats.

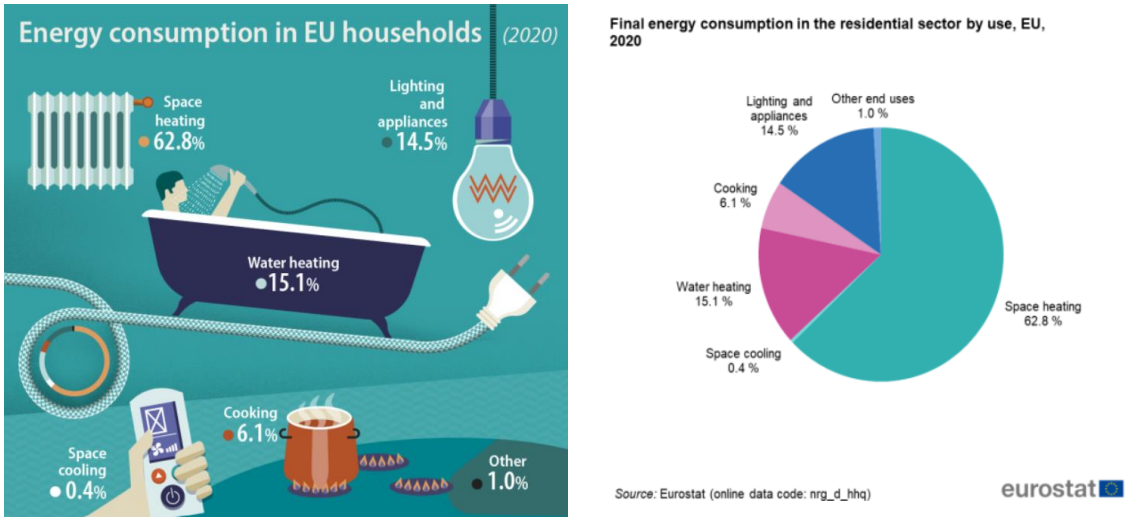


Figure 0.1: Energy consumption in EU households

Let's talk about better comfort, and how to create an environment with better comfort. Well, this means improving the atmosphere of the occupant, but how does controlling temperature lead to better comfort? The factors that play an important part in creating a better environment are multiple like humidity, quality of air, and brightness, but the most and far important factor is temperature control. There is not any specified temperature that has better comfort but it's the user is the one who defines the temperature, he/she feels good about. The temperature user defines in the thermoset is called the set point and the best temperature control algorithm is the one that controls the temperature closest as possible to the set point.

The conventional ON/OFF thermostat has been used for a long time, but it comes with some drawbacks that are not noticed in the past. But now it cannot be ignored anymore. It has high-temperature swings, that lead to low comfort and high energy consumption. Moreover, the number of ON/OFF cycles leads to wear and tear for the heating device and lowers the life of the relay and battery of the thermostat as well.

All these drawbacks of the ON/OFF thermostat led to the development of this project this will have less temperature swings, less wear and tear for the heating device, and

improved life of the relay and battery in the thermoset. These are the aim of the project to develop Time proportional integral (TPI) temperature control.

Figure 0.1: Thermostat by Fantini Cosmi (Final product)

1 State of the Art

In this chapter we will try to make a basic concept of the project. Staring point would be understanding the working principle of thermostat or temperature control, the simplest model using block diagrams, configuration of a feedback temperature control system, different types of temperature control systems in the market. Furthermore, finally time proportional control (TPI) that is to be implemented which is our ultimate goal of this project.

In this project, the main aim is to implement the TPI (Time proportional integral) temperature control. However, it is better to know all the temperature controls that now a days taken into consideration. Undertaking and comparing each would provide clearer image of pros and cons of each temperature controls methodologies.

In this chapter also remark and captured the basic configurations of the system of the system.

1.1 Basic Configuration for Temperature Control

The device which is used to control temperature is called as “thermostat”. The basic hardware component it has been

- Temperature sensor
- Microcontroller
- Relay

1.1.1. Temperature Sensor

It’s hardware that has thermocouples that senses the temperature in physical form where it is installed. It converts the temperature to a quantity like voltage or resistance in digital form and feedbacks the values to the microcontroller.

1.1.2. Mircrocontroller

It is considered the brain of the thermostat, it is the hardware part where all the logic and algorithms are stored in. It receives the temperature values from the temperature sensor and after careful calculation, it commands the output to relay.

1.1.3. Relay

Its is the componet that gives controller command to on and off switch.

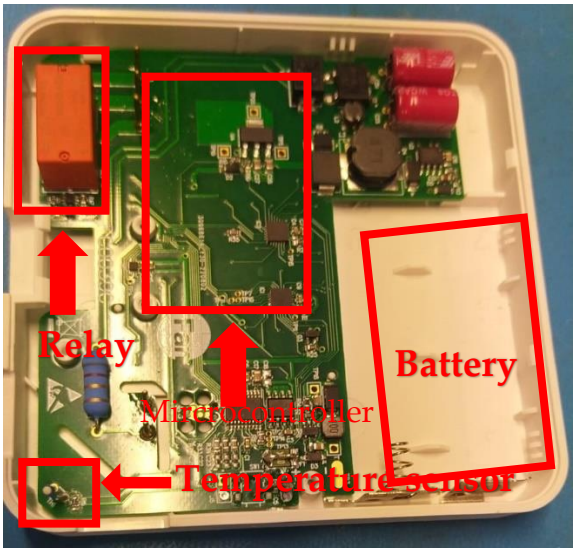


Figure 1.1: Fantini Cosmi Thermostat's hardware components

1.2. Temperature Controller Principle

The following figure 1.2 shows an example of a feedback control system used for temperature control. The major parts of the feedback control system are built in the temperature Controller. A feedback control system can be built, and temperature can be controlled by combining a temperature controller with a controller and temperature sensor that are suitable for the controlled object.

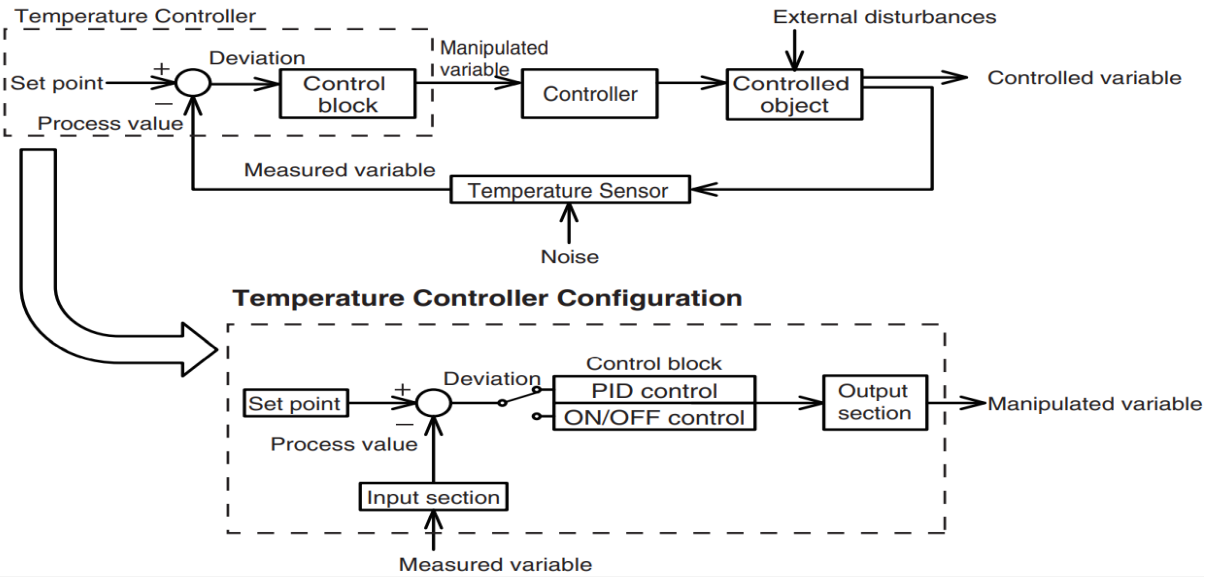


Figure 1.2: Temperature control principle / Configuration of a Feedback Control System

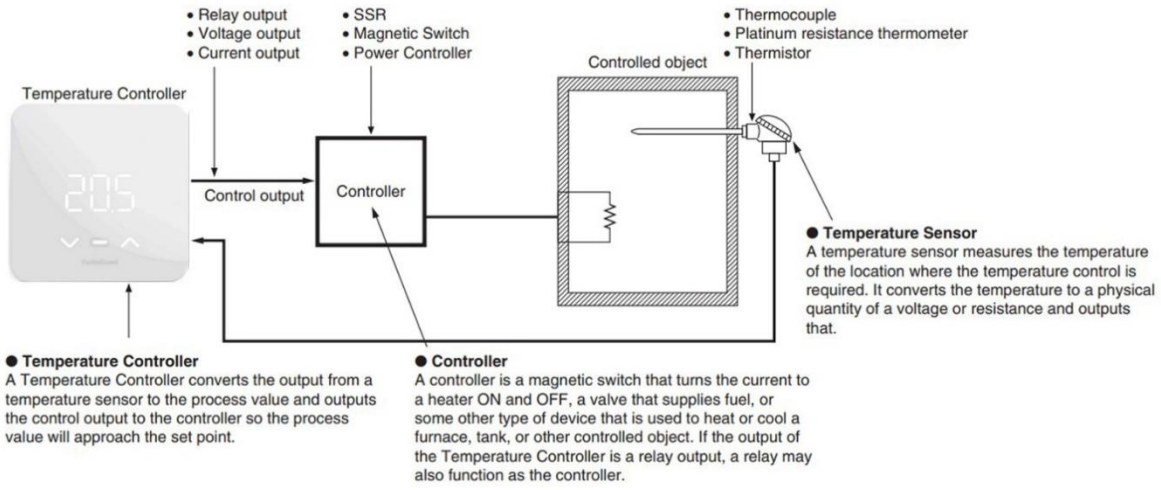


Figure 1.3: Temperature Control Configuration Example

1.3. Types of Temperature Control

There are a lot of temperature controls under study and under implementation, the two broad categories of temperature control are.

- Standard Temperature Controls / ON/OFF control
- Advanced controls

1.3.1. Standard Temperature Controls / ON/OFF Control

Before going into further details need to know about basic definition or terminologies.

1.3.1.1. Definitions

Set Point/ desired value: is the temperature that is set by the user in thermostat that is desired by him/her.

Temperature differential: The temperature difference between maximum temperature swing and minimum temperature swing from set pint.

Overshoot: The temperature deference between maximum temperature and the upper set point in a ON/OFF cycle OR the temperature higher than the temperature differential.

Undershoot: the temperature deference between minimum temperature and the lower set point in a ON/OFF cycle OR the temperature lower than the temperature differential.

ON/OFF is the simplest temperature control, it is not only long served temperature control, but it is still in use because of its simplicity and robustness. However, now it's not considered convenient to use because of poor comfort and high energy cost.

Perhaps the basic principle of ON/OFF control would give a better understanding of the cons associated with this kind of temperature control. The switch of the heating device (Boiler, heat Pump) is turned OFF when the temperature sensor reaches the Upper set point and when reaches the lower set point it makes the heating device ON. The temperature continuously fluctuating between these two set points would cause wear and tear on the heating device and relay of the controller.

Moreover, the temperature differential set for ON/OFF control is usually (2-3 °C) to prevent excessing ON/OFF cycles which would ultimately cause wear and tear, but a higher temperature differential will give poor comfort. The problem with a high-temperature differential is that each degree Celsius away from the set point (desired temperature) cost us unwanted energy lost and hence difference can be seen on your electricity bill or gas bills. Furthermore, the problem with ON/OFF controls is due to thermal inertia Experience overshoot and undershooting.

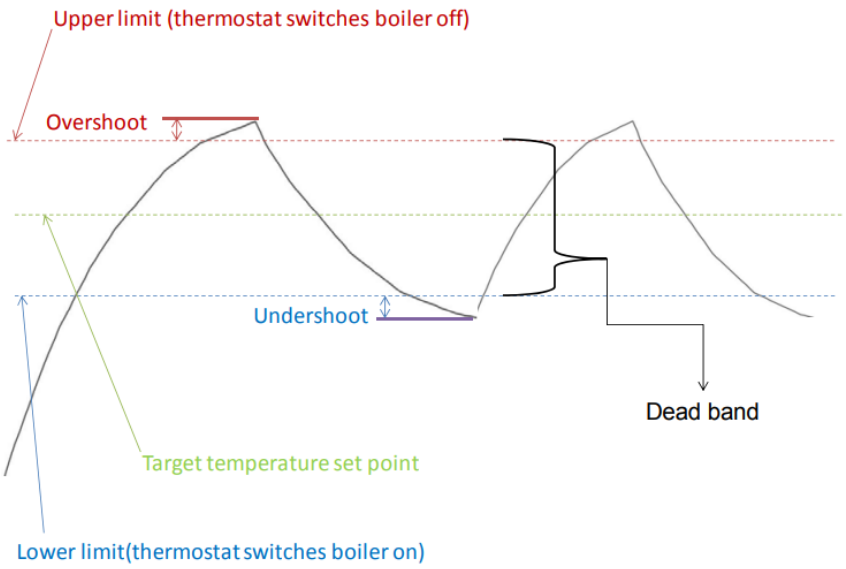


Figure 1.4: Standard Temperature Controls / ON/OFF control temperature response point out overshoot and undershoot

1.3.2. Advanced Controls

The advanced temperature control has ability to control temperature more accurately and achieve better energy efficiency compared to ON/OFF or standard temperature control.

The most common advanced temperature controls are

- Weather compensation control
- Load compensation control
- Time proportional control (TPI) Control

All these controls have one thing in common, they can control heat output accordance with the heat demand, therefore they are considered more efficient as far as temperature accuracy and energy efficiency is concerned.

1.3.2.1. Weather Compensation Control

Weather Compensation [3] controls are one kind of proportional control that estimate the heat demand based on the external temperature. It involves the recording of temperature-by-temperature sensors installed outside of the building.

The boiler receives this signal from the temperature sensor and modulates the head demand accordingly. Addition saving is expected from weather compensation control compared to standard one by lowering the return temperature of the boiler when the external temperature is lower and vice versa.

As boiler efficiency is better when it operates at a lower returning temperature. Weather compensation control exploits this property to improve energy efficiency.

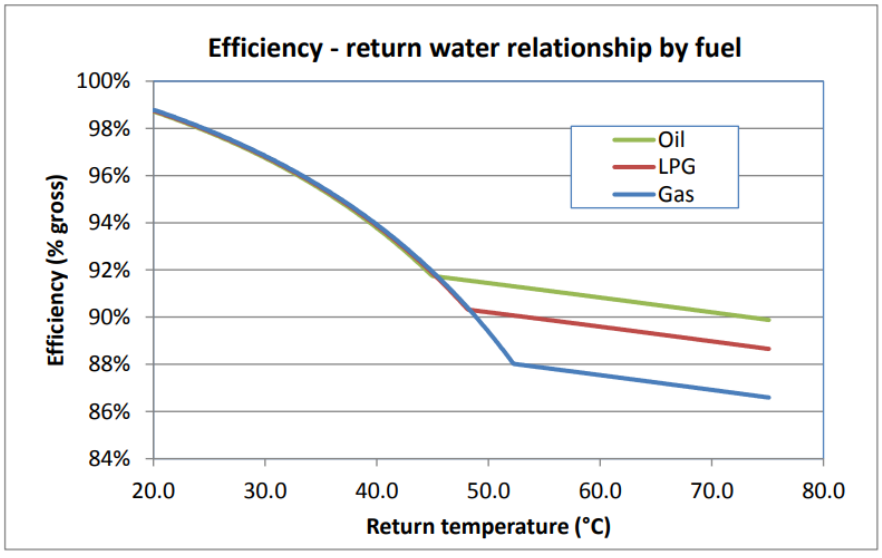


Figure 1.5: Theoretical relationship between boiler efficiency and return water temperature by fuel type

The relation between efficiency and return temperature shows that efficiency is better when the return temperature is lower for all fuels.

Moreover, as we operate at lower temperatures, we have lower thermal inertia and hence low overshoot which was on of the cons associated with standard ON/OFF control. Therefore, compensation control provides greater comfort.

The simple form of weather control uses a single curve (proportional control only) to define the relationship between boiler flow temperature and outside air temperature. It's usually an open-look control and it must be tuned to ensure desired results.

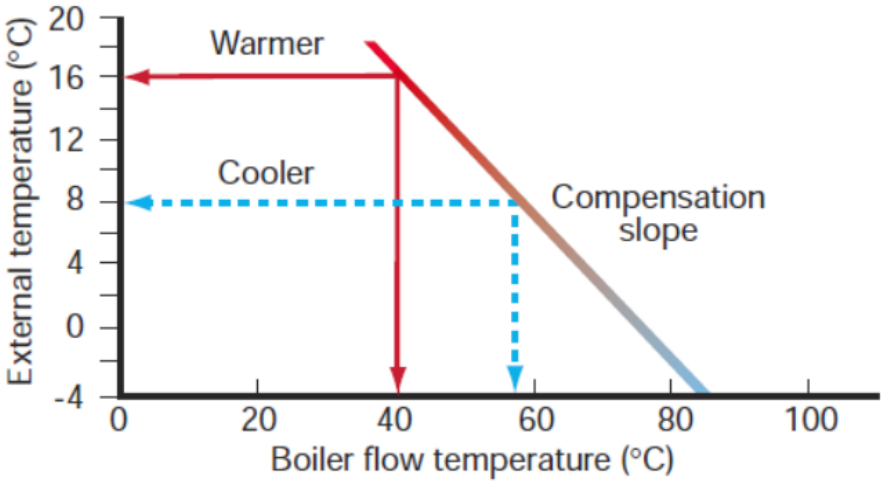


Figure 1.6: Simplified weather compensation control curve

1.3.2.2. Load Compensation Control

The basic principle of load compensation control is very much like weather compensation control with only difference that sensor is installed inside the building instead of outside. It measures the heat demand based on feedback from temperature sensor.

1.4. Time Proportional and Integral (TPI) Control

Time Proportional and Integral (TPI) Control is the main aim of this project. Contrary to Simple ON/OFF control which only operates around a set point, Time Proportional and Integral (TPI) control operates on a time and temperature basis, it uses algorithms that calculate the expected heat demand on basis of time taken to achieve set points in the previous cycle. It predicts how far room temperature is from the set point/ desired value, and on the base of that, it calculates how long the boiler should be turned ON to reach the desired temperature.

Time Proportional and Integral (TPI) [2] control operates only once the temperature falls within the proportional band which is usually 1-2 °C around the set point. The concept of the proportional band is explained in chapter 5. In TPI controls we define a cycle period of 5 minutes to 15 minutes depending on the heating system we design our controller for, within this cycle period TPI control calculates the ON and OFF ratio that would be proportional to the difference between the room temperature and the set point. In simple words ON and OFF time ratio depends on the heating demand of the room.

Perhaps an example would be convenient to understand. In 5 minutes, cycle period if the controller gives us 40 % output after calculating the heat load, in response the boiler/heating device will turn ON for 2 minutes and Off for 3 minutes. Let's say the set point is 20 °C and the proportional band is 2 °C then the proportional band range will be from 19 °C to 21 °C, the logic implies that of the room temperature is more than 21 °C then the next cycles period will be complete OFF (5 minutes) and if the room temperature is lower then 19 °C cycles will be complete ON (5 minutes). Finally, if the room temperature falls within the proportional band, it will apply an ON/OFF ratio in accordance with controller calculations.

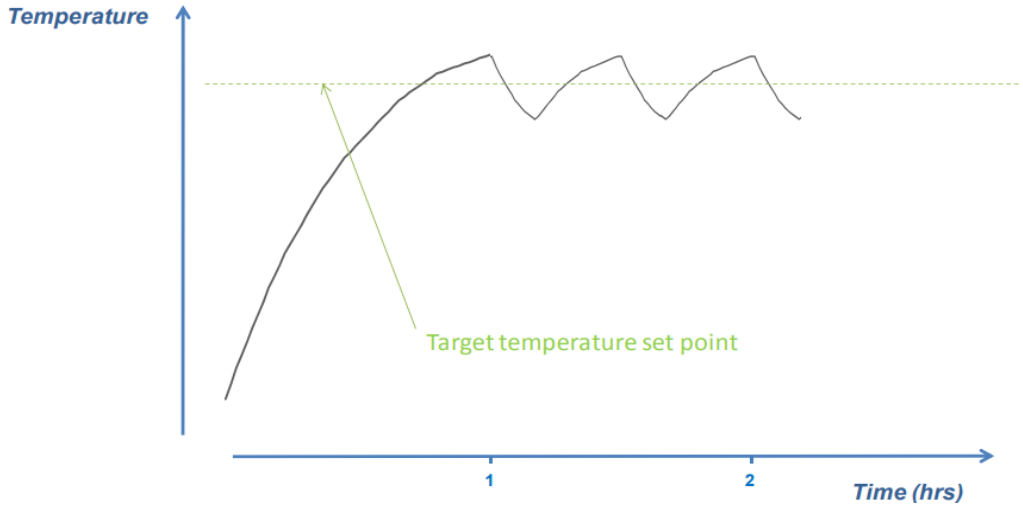


Figure 1.7: Standard ON/OFF control temperature profile

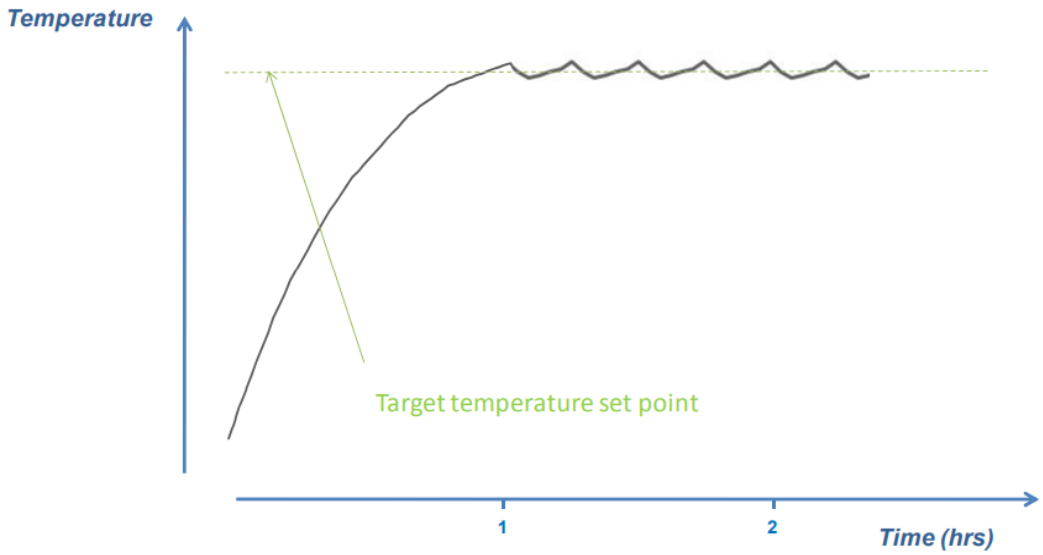


Figure 1.8: Time proportional integral (TPI) temperature profile

1.4.1. Advantages and Risks Associated with TPI Control

Advantages

- More stable temperature control hence better comfort compared to standard thermostat
- More stable temperature means less temperature fluctuation around the set point. This could potentially lead to sustainable energy savings as even small decrease in average room temperature will leads to notable reduction in heat demand and therefore the energy consumption.
- Improving overall efficiency of heating system.

Risk

- It is believed that TPI will cause more wear and tear on boiler, but there is no evidence of that.

1.5. Heating Systems

1.5.1. Radiator

Figure 1.9: Radiator

1.5.2. Fan Coil

Figure 1.10: Fan coil

2 Targets of the Project

In this chapter, we will explain the goal of the project which is to develop a thermostat but it's not easy as it seems. Therefore, first, we set the goals of the project and define the road map of the project in this chapter. First, the brief comparison between the conventional ON/OFF thermostat and TPI which is the aim of the project to develop will be explained. Based on the benefits of the TPI temperature-controlled thermostat over the ON/OFF control thermostat will give us a clearer pathway to our goals.

Moreover, in this chapter, development procedures and the scope of work will be explained step-by-step so as to get a better understanding of the project. Not only the ultimate goals are explained in this chapter, but also the steps that we follow be explained e.g., development of Simulink model, control design, etc. considering these our interval targets.

2.1. ON/OFF vs TPI Controlled Thermostat

The comparison between ON/OFF and TPI-controlled thermostats gives us a better understanding of the project and helps us determine the goals we should set. Conventional thermostats are long been in use but have some cons that are not efficient for nowadays market. The main problem they possess is the high-temperature swings which mean using more energy. Moreover, high swings also lead to poor Comfort. Therefore, new temperature control algorithms should replace conventional ones in order to have better comfort and high efficiency.

In comparison the conventional ON/OFF thermostat has a lot of disadvantages, to overcome all those disadvantages TPI controlled thermostat comes into play, which improves the efficiency of heating systems (radiator, fan coil), provides better comfort, increases the life of boiler and thermostat itself. In Table.2-1 the comparison is mentioned.

Factors	Conventional ON/OFF	TPI
Temperature differential/swing	2-3 °C	± 0.6 °C
ON/OFF cycle per unit time	High	Low
Overshoot / undershoot	High	Low
Comfort	Low	High
Efficiency of heat system	Low	High
Lift time of relay (depends on ON/OFF)	Low	High
Battery life of thermostat	Low	High

Table 2.1: Comparison of conventional ON/OFF & TPI controlled thermostats

2.2. Targets of the Project

2.2.1. Reduce the Differential/Swings of Temperature around Set Point

As mentioned before, the goal is to have low as possible temperature swings around the set point because as we have 1°C degree temperature more than the set point that means we are using undesired energy. Hence leading to low efficiency. Moreover, a high-temperature differential also means low comfort as well, so the temperature differential needs to be kept low as possible.

As far as this project is considered company has set the tolerance value of temperature swings or temperature differential is ±0.6°C around the set point. On the contrary, in conventional thermostats, it's usually 2-3 °C.

2.2.2. Approach the Set Point as Fast as Possible

This is an important aspect of the project; the temperature control algorithm should be able to approach the set point as fast as possible. It's logical that customers can not wait for hours to get the desired temperature in the room. For an instant, if the set point is 20 °C and the current temperature of the room is 12 °C, we can't our control algorithm to approach the set point for at least the time possible.

However, approaching the set point fast as possible also depends on the heating capacity of the heating system and the load of the systemin. For example, if we have a powerful fan coil it will approach set posit fast compared to a less powerful one, on the on hand, if we are using a fan coil with the same power but our room is bigger, or

transmittance of the wall is high or outside temperature is low that all lead it high heat loss that means high load. therefore, with a high load, it takes a higher time to approach the set point.

Nevertheless, all these factors like heating capacity and load play a vital part but the best temperature control algorithms are the one that utilizes these factors in the best way possible.

2.2.3. Reduce the No. of ON/OFF Cycle as Low as Possible

Reducing the number of ON/OFF is important as this affect the life of the heating equipment and thermostat itself. As far as the life of the thermoset is concerned, the life of the thermoset depends on the life of the relay in it. the relay has a limited number of ON/OFF cycles in its life period. Therefore, if the temperature control algorithm is making it turn on and off more that will shorten the life of the relay and hence thermostat.

Moreover, the battery consumption also depends on the ON/OFF cycle if the relay needs to turn on and off more, the battery will consumer more and hence need to be replaced more often.

Furthermore, if heating equipment is to make turn on and off more it will cause wear and tear to the equipment which will reduce its life of heating equipment. So, it relays important to take into account this aspect as well in our project. However, it is expected that in Time proportional temperature control (TPI) is relatively more than the conventional ON/OFF, but no studies prove this perception.

2.2.4. Should be Compatible for all the Energy Class

As far as this project is concerned, the thermostat should be able to work in each energy class. It is related to control design in which we need to tune it according to each energy class. Each energy class is defined according to its transmittance, with each transmittance our controller will gain value.

On the other hand, a controller can be designed which will access the heating capacity and load of the system and act accordingly just like a dynamic system. you don't have to define your energy class, just install and I will automatically calculate the energy needed and try to control the temperature. Both of the methods will be explained in the next chapter control design at length.

2.2.5. Saving Energy (10 % More Than Conventional Thermostat)

In this project, it is our goal to prove that time proportional control (TPI) controlled thermostats save you 10 % more energy than the conventional thermostat that we have found in the studies. After implementation of the project, we will try to prove the fact that time proportional control (TPI) controlled thermostats are better than ON/OFF thermostats as they save you 10 % more energy.

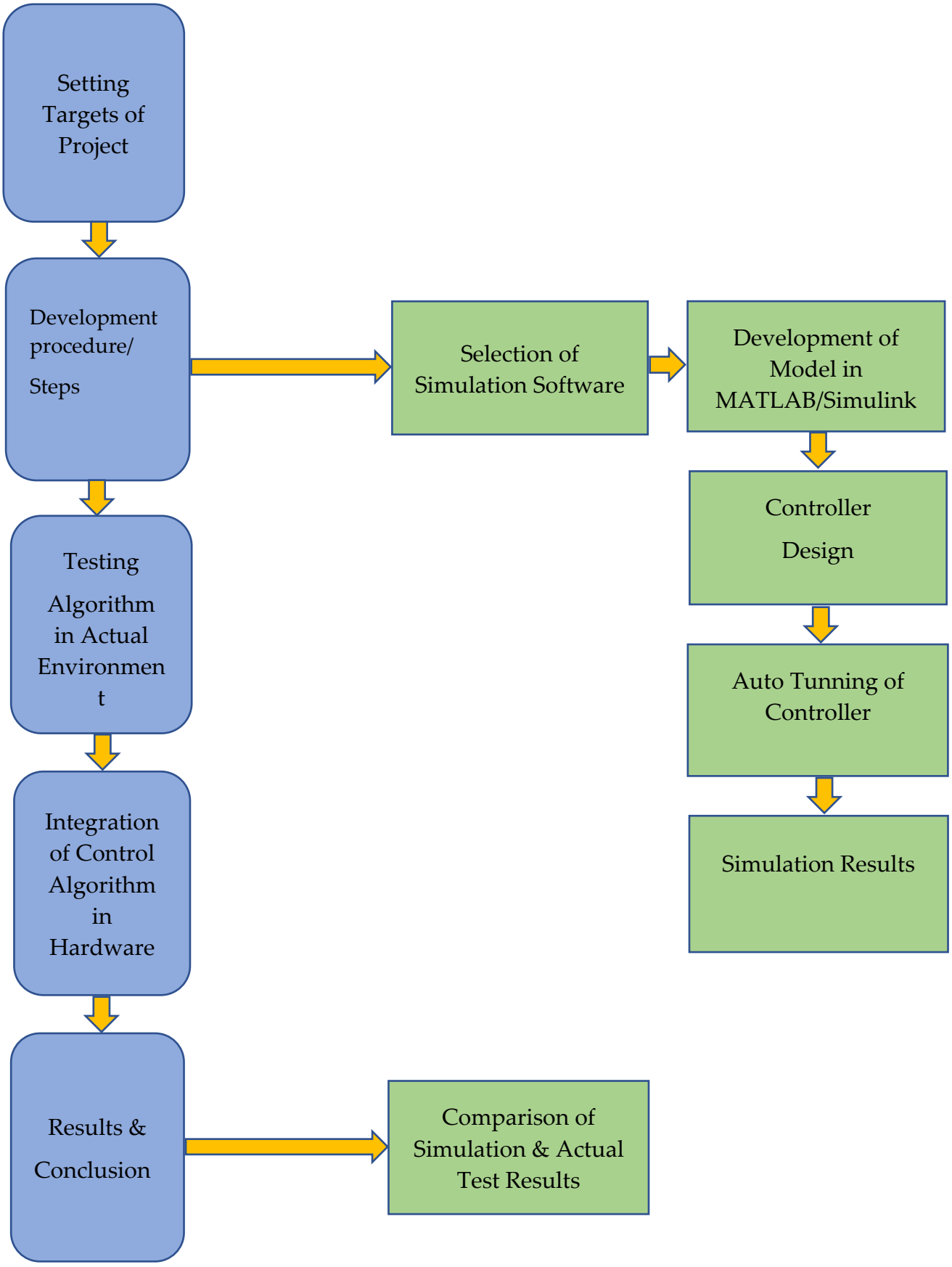
2.2.6. Thermostat Should be Compatible for Radiator & Fan Coil

As per Fantini Cosmi S.p.a requirements, most of the focus is to develop temperature control algorithms that are best suited for Radiator & fan coils, as they also sell these products and want to develop the thermostat which is best compatible with their own heating devices. Considering their own marketing strategies

Factors	Target	Comment
Differential/Swings of Temperature	±0.6 °C	±0.6 °C Around set point
Approach the set	FAST	Fast as possible
ON/OFF	Low	Turn ON & OFF should be low as possible
Energy class	For all	Should be compatible for all Energy class
Energy efficiency	10 %	10 % better efficiency then conventional ON/OFF thermostat
Heating equipment	Radiator & Fan coil	Should work with convenient temperature accuracy for radiator and Fan coil

Table 2.2: Targets of the project

2.3. Development Procedure/Steps



2.3.1. Software Requirement

We are living in a modern world, as it's not that old time when devices were developed in an actual environment and devices were tested again and again until reaches desired results. Nowadays software comes into play where we can develop devices and only test when we have desired results obtained after simulation in software.

Therefore, after setting our goals and targets, it's pointed to decide which simulation software is best suited for our project, there were no better than the software, which is explained below,

- MATLAB
- Simulink
- Design builder

2.3.1.1. MATLAB

MATLAB is a software developed by MathWorks that provides a programming platform for complex applied mathematics with an extensive mathematical library with graphical representation allowing precisive analyses for engineers, developers, and researchers. It offers a substantial benefit over any other programming language as it is easier to understand and interchangeable with other languages such as C, Turbo Pascal, and Fortran. MATLAB has many specific toolboxes concerned with certain fields, having numerous tools to design, solve and resolve complex issues through command or programming mode with graphical representation. It is the leading software for mathematical depiction and understanding signal processing, automation, neural networks, structural computations, and statistics.

2.3.1.2. Simulink

Simulink software represents MATLAB files through graphical representation allowing users to analyze mathematical functions in real-time, giving authority for modeling new structures and simulating them for better understanding. Simulink makes mathematical modeling easier by allowing users to give visual input and judge the system based on their output. Simulink authorizes designing nonlinear systems, which is considerably strenuous otherwise. This software gets used in deep machine learning, technical and statical computation, control systems, signal processing, and many other technical fields.

2.3.2. Scope of Work/ Hierarchy of work

It's time to get serious in work, after setting project goals and done with, which software we are going to use it's time to set-up the scope of work. Hierarchy of work is the step-by-step procedure we are going to follow.

- Develop model in MATLAB/ Simulink where we can simulate our model and achieve the desired goals/results
- Test the model in design builder in order to validate our model and TPI algorithm.
- Integrate Simulink model into hardware 4. Testing the thermostat in actual environment

3 Modulization of the System & It's Components

This chapter analyzed the physical model of our system and its components. The model and its components will manifest the actual physical phenomena. For this purpose of developing a model that represents the actual system, the company decides to use MATLAB-Simulink for developing, programming and initially testing the temperature control algorithm.

Furthermore, this chapter represents the step-step process to develop different components of the model in Simulink and the respective methodical theory behind it, and the different parameters that represent the component. Hence, the whole modulization of the system.

The reason for the priority of the company, more focus was on the heating part, and cooling was overshadowed. The model used is for the Fan coil and radiator as a heating device.

3.1. Room/ Building Zone

Since on Simulink it was necessary to create a system on which to model the algorithm, for the modeling of the room examined so far it was necessary to define a linear time-invariant system that was found in the textbook [12]. The system consists of walls with a UA [W/K] (called k in the equations) and a resistance that generates power in the interior room

After finding the mathematical Model of the room, it is then represented in Simulink by using its "state-space" block

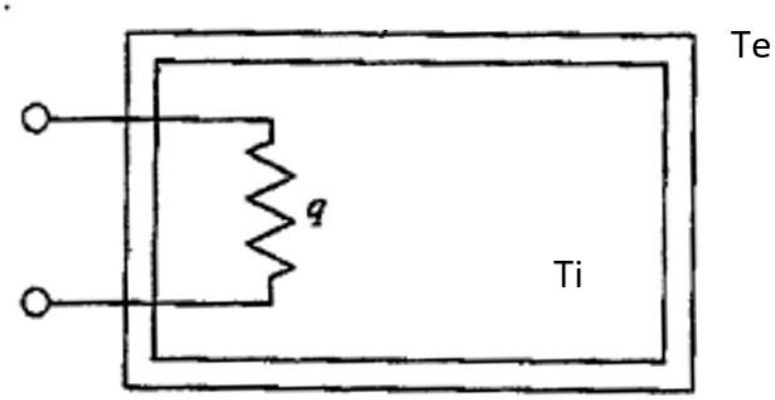


Figure 3.1: Schematization of room model with internal generation q, internal temperature Ti and external temperature Te

Variable	Description	SI unit
C	Heat capacity of the room (part of the wall + air)	J/k
k	*UA* of the walls	W/(K)
U	The Transmittance of the walls	W/(m 2 ·k
A	Area of the room	m2
$T_e(t)$	Manipulable variable (Disturbance) external input temperature →$u_1(t)$	k(°C)
$q(t)$	Power supplied to the manipulable variable input system →$u_2(t)$	kWh
$T_i(t)$	Variable internal temperature status →$x(t)$	k(°C)

Table 3.1: Parameters used in the dynamic model of the room

This model is highly simplified compared to that of Design Builder, but for reasons of time it was necessary to adapt.

The equation of state is derived from an energy balance of the system:

$$C\dot{T}_i = k\big(T_e(t) - T_i(t)\big) + q(t)$$

Using the state representation form the system can be written as follows:

$$\dot{T}_i = -\frac{k}{C}T_i(t) + \frac{k}{C}T_e(t) + \frac{1}{C}q(t)$$

$$y = T_i(t)$$

From which the matrices of the A, B, C and D system are derived:

$$A = \left[-\frac{k}{C}\right] \qquad B = \left[\frac{k}{C} \quad \frac{1}{C}\right]$$

$$C = [1] \qquad D = [0 \quad 0]$$

Which must be inserted in the "**state-space**" block of Simulink. The inputs to the "**state-space**" block are the external disturbance which is the external temperature and the other is the power input/ Heat delivered (q(T)) to the room in order to control the temperature of the room.

The external disturbance could be a constant temperature e.g. 0 °C (273K) if the simulation time considered is really short or can use a sine wave of 24 hours, with temperature fluctuation of - 10°C to +10°C. Both blocks are shown in figure.3-4. On the other hand, the power q(t) to be delivered is decided by the controller. The

temperature of the room (Toom) will be the current temperature sensed by the temperature sensor inside the thermostat which would be feedback to the controller.

In addition, in the "**state-space**" block it is worth mentioning that the "Initial conditions" represent the staring temperature to be mentioned and "Sample Time" as well, which represents the frequency of temperature measurement by a temperature sensor (60 seconds in our case).

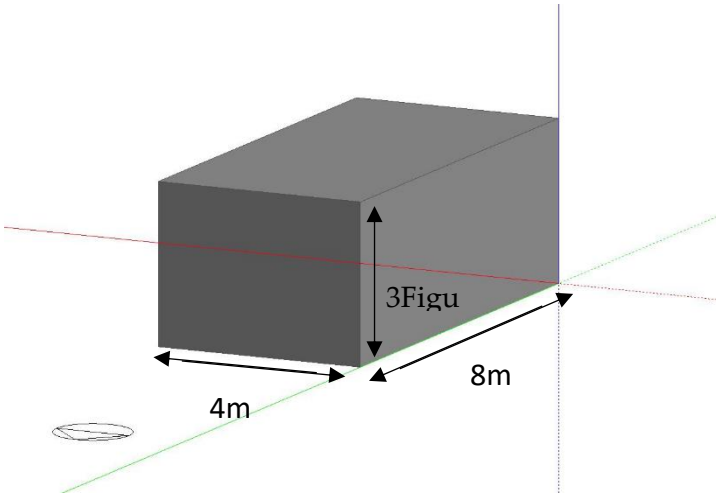


Figure 3.3 : Dimension of the room for simulation

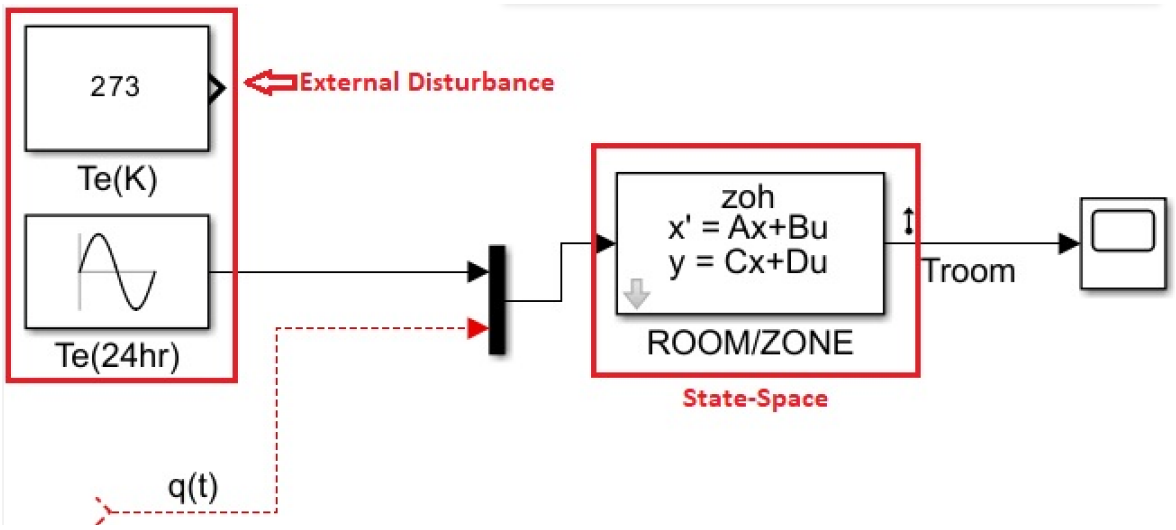


Figure 3.4 : Room and disturbance Simulink block

3.2. Heating Systems

For simulation purposes and company's interest is to develop temperature control algorithms for

- Radiator
- Fan coil

Therefore, it was required to build Simulink model in for radiator and fan coil as well.

The mathematical model of fan coil and radiator are as follows

Fan-Coil: $\dot{q} = \dot{m}_{H2O} \cdot c_{H2O}(T_{H2O} - T_i)$

Radiator: $\dot{q} = A_r \cdot \sigma \cdot \varepsilon\left(T_{H2O}^4 - T_i^4\right)$`

Variable	Description	SI unit
$q(t)$	Power supplied to the manipulable variable input system →$u_2(t)$	kWh
T_{H2O}	The maximum temperature delivered by the heating device	k(°C)
T_i	Variable internal temperature status/Current room temperature	k(°C)
c_{H2O}	Specific heat of water	J(kg⁻·K)
$\dot{m}_{H2O}$	Mass flow rate of circulating water	m3/s

Table 3.2: Parameters used in the dynamic model of the heating systems

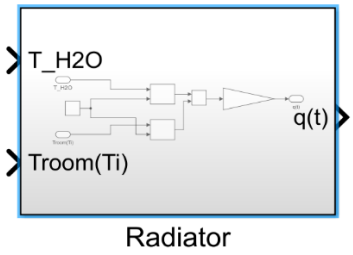


Figure 3.5 : Simulink subsystem for radiator

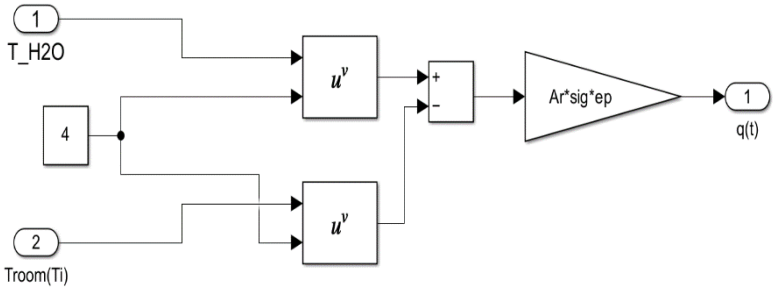


Figure 3.6 : Simulink design of radiator

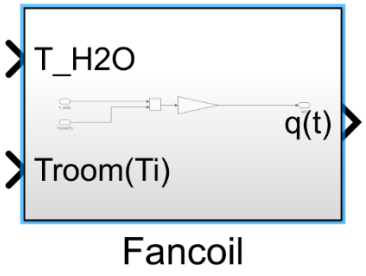


Figure 3.7 : Simulink subsystem for radiator

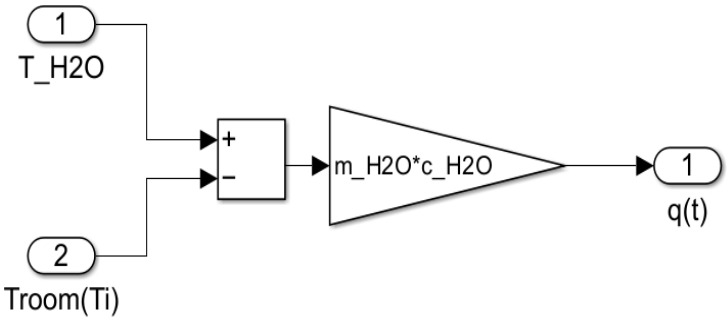


Figure 3.8: Simulink design of radiator

3.3. The Switch

The controller give output in form of 1 or 0 which represents ON and OFF respectively. The switch is designed using built-in block of "switch" which has if else logic in it. If controller output is 1 for certain time, then the switch will be working with full capacity delivering (T_{H2O}) to the room, on the other hand, if output is 0 that means OFF state, Troom (Ti) will be considered.

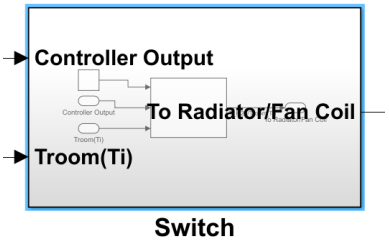


Figure 3.9: Simulink subsystem for heater switch

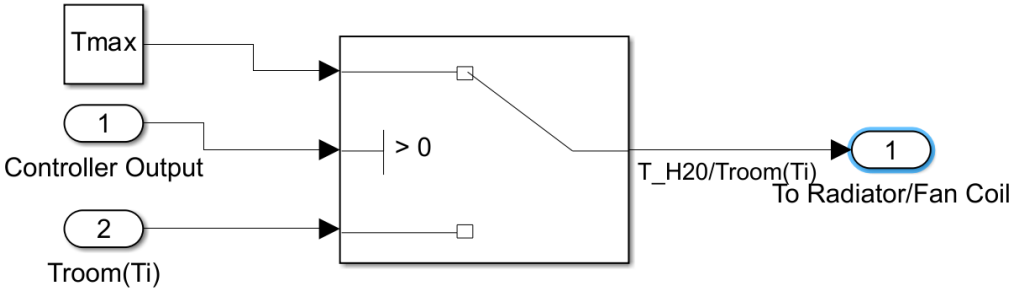


Figure 3.10: Simulink design of heating switch

3.4. PWM (Pulse Width Modulation)

Pulse Width Modulation is a technique that is really handy and widely used in various applications, especially in power electrocnics, telecommunications, etc. Pulse Width Modulation is a way of delivering signals in pulses rather than continuously varying analog signals. In simple words, it converts the continuous analog signal into ON and OFF signals with respect to time, if the analog signals have a higher deviation then the output signal will have a higher ON time period.

Pulse Width Modulation output signal has two most important components which define its behavior: frequency and duty cycle(D). Duty cycle(D) is the time period in which signals are ON (high) with respect to the total time period of the cycle (ON time + OFF period) in percentage form.

$$Duty\ cycle = \frac{ON\ time}{Cycle\ period} \times 100\ \%$$

The above relation defines the duty cycle

$$V_{AVG} = DV_H + (1 - D)V_L$$

Describes the average signal

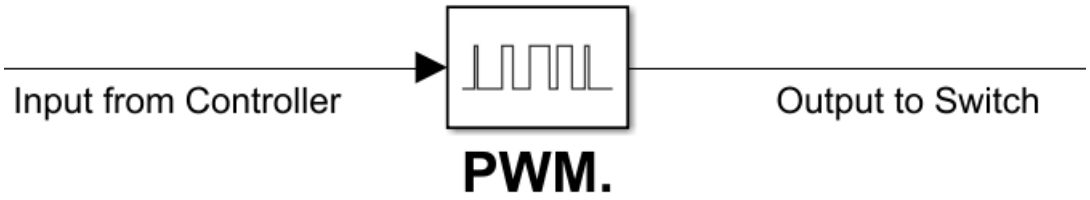


Figure 3.11: Simulink block for Pulse Width Modulation (PWM)

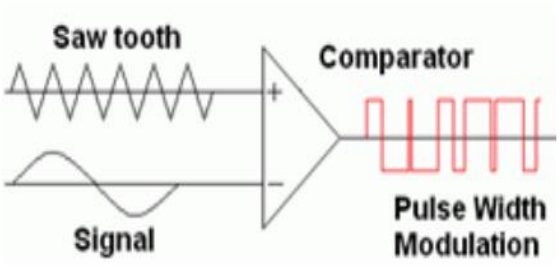


Figure 3.13 : Principle of PWM

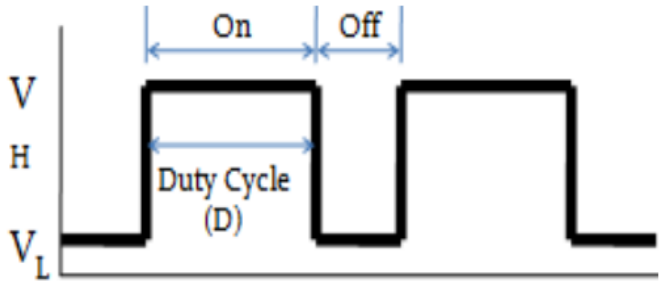


Figure 3.12- PWM Signal

3.5. Controller

"The controller development will be explained in next chapter as it's our main GOAL To design"

3.6. The Full Model

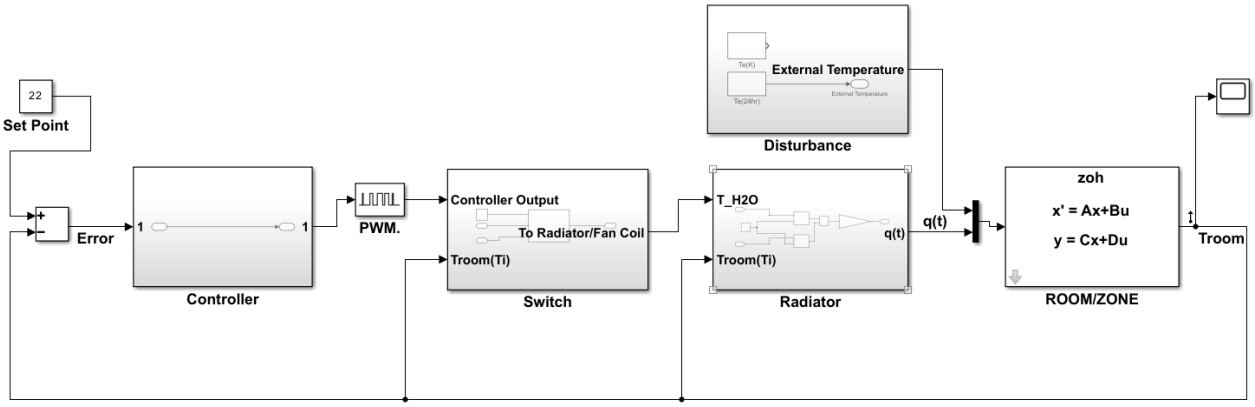


Figure 3.14: Basic Simulink model for temperature control

4 Control's Theory and Characteristics of Our System

4.1. Control Design Theory

Before diving into theory of controls, better to understand some common controls terminologies.

4.1.1. Control Terminology

4.1.1.1. Offset:

Usually in case of proportional control action, a permanent error is left between process variable and the set point. That error is called offset. Offset is basically difference between process variable and set point. It may be above of below the set point as shown in figure 4.1.

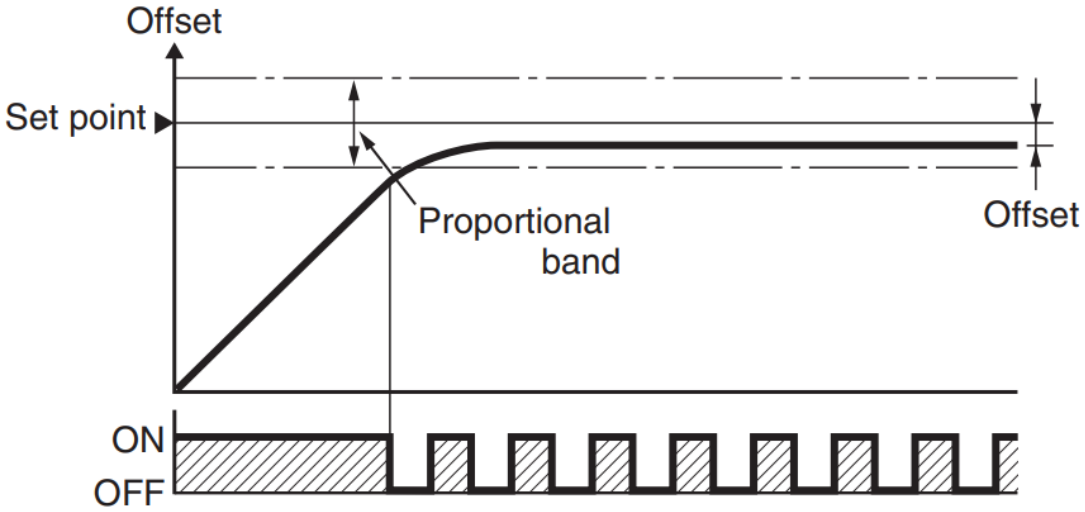


Figure 4.1: Representation of Off-set & Proportional band

4.1.1.2. Hunting and Overshooting

ON/OFF control gives us the waveform shown in figure 4.2, due to the thermal inertia of the system, the usual temperature exceeds the set point. That rise in temperature is called Overshooting. The temperature oscillation around the set point is called Hunting. The best temperature control is expected to reduce both the Hunting and overshoot as low as possible.

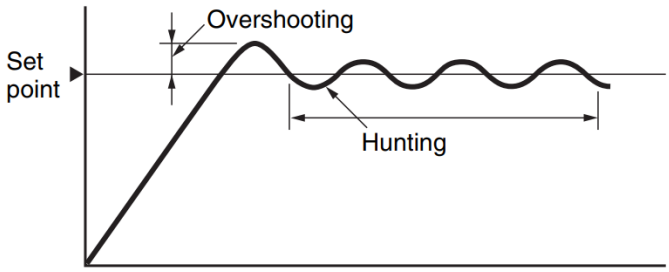


Figure 4.2: Graphical representation of Overshooting and hunting [4]

4.1.2. Types/Methods of Control

- ON/OFF control
- P/ proportional control/band/ control
- PI control
- PID control

4.1.2.1. ON/OFF Control

The ON/OFF control is the simplest, most handy, and has long been in use in the control world. If the present values in lower than the set point, the out of the controller will be ON which means the heater will supply power until it reaches the set point plus 1-2.5 degrees. Once it reaches this point, it will turn OFF means the heater is shut off. Once the system reaches a set point minus 1 - 2.5 °C it will turn ON again and the cycle goes on unless you change the set point.

In ON/OFF control, the output of the controller is either ON or OFF based on a set point, as it fluctuant around two points the ON/OFF controller is also called two-position control action or Bang -bang control.

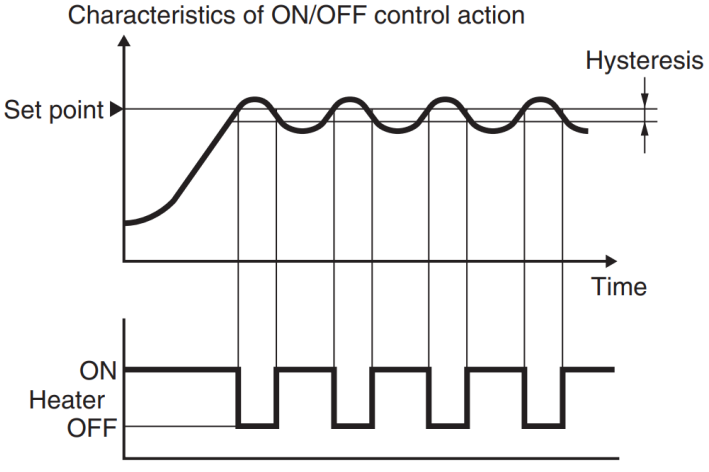


Figure 4.3: Graphical representation of ON/OFF temperature control

4.1.2.2. P Action (Proportional Control Action)

P action or proportional action control adjust the output of control is in proportional to the change in process values (temperature) with the proportional band. The proportional band sets the upper and lower limit and 0% and 100% represents the upper and lower limit of the proportional band respectively.

The rules p only controls follow:-

- The process variable (Temperature) that is proportional to deviation in output only then the process variable falls with the proportional band.
- A 0 % Output when the process values is higher than the proportional band
- A 100 % Output when the process values is lower than the proportional band

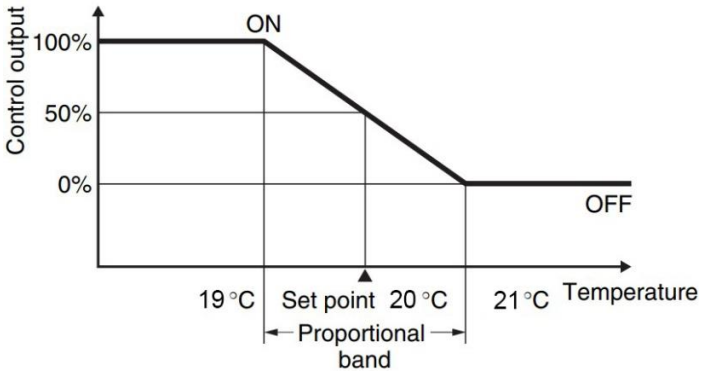


Figure 4.4: Principle of proportional band & P only control

Setting proportional band in P only control is critical as if the proportional band is too wide, we are left with Hunting that means we may never reach Set point. On the other hand, if proportional band is too narrow, we may experience oscillating results that make it no better than ON/OFF

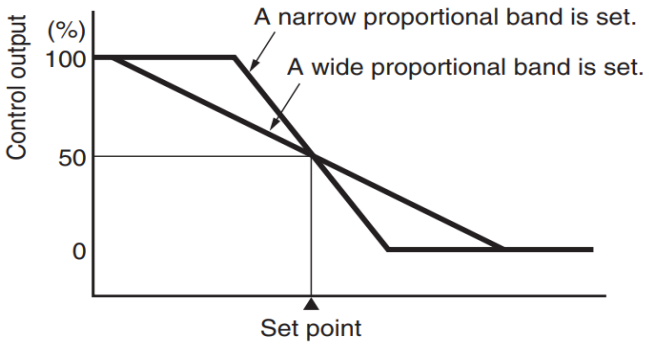


Figure 4.5: Demonstration of narrow and wider proportional band

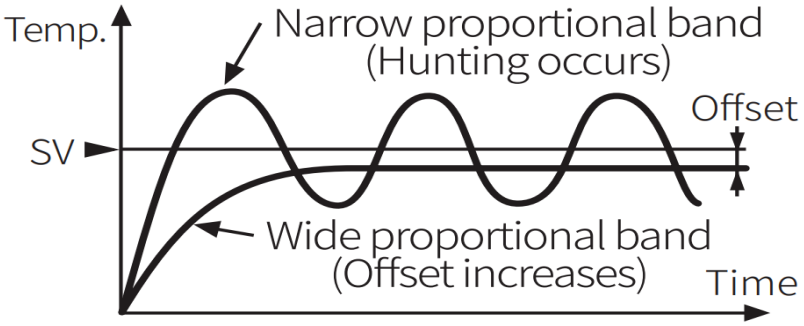


Figure 4.6: Results comparison of narrow and wider proportional band

4.1.2.3. I Action (Integral Control Action)

To remedy the problem with P control only, which is to be left with Offset. The integral action comes to the rescue, the integral action automatically adjusts the offset. The integral action tries to approach the set point and reduce the Offset that is usually left with P control only.

However, integral action along with proportional will take a long time to stabilize the temperature. Integral action cannot operate all by itself, it requires at least proportional control with it.

If the integral action is defined in time, if the integral action is long, it will take longer to remove offset. On the contrary, if integral time is less then it will approach the set point fast but cause hunting.

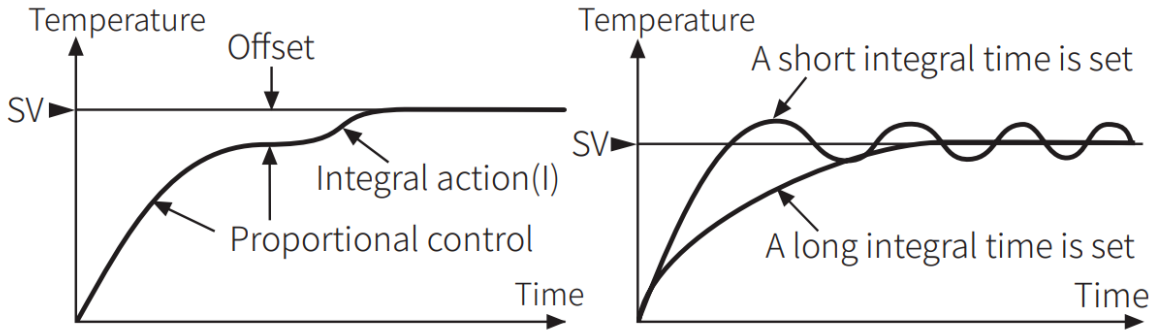


Figure 4.7: Effect of integral action on temperature control & comparison of shorter integral time and longer integral time [10]

4.1.2.4. D Action (Derivative Control Action)

So far proportional and integral action both can control the temperature around a set point but under external disturbance will give you inconvenient control. Therefore, here derivation control comes into action as it overcomes the external disturbance

effect.

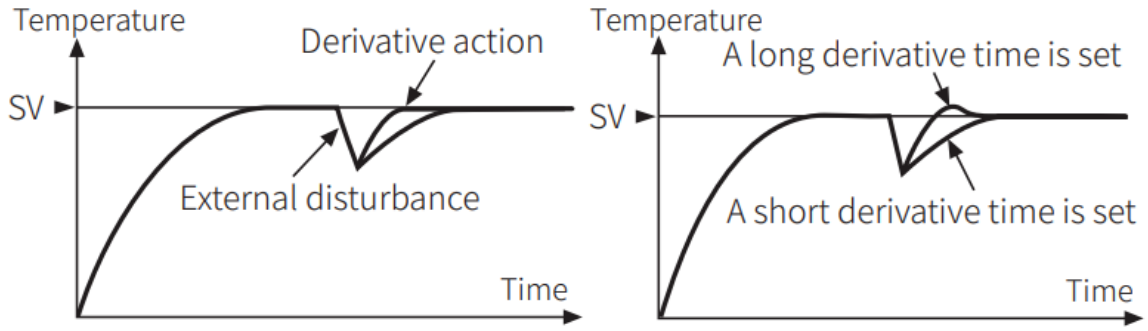


Figure 4.8: Effect of derivative action on temperature control & comparison of shorter derivative time and longer derivative time

4.1.3. PID control

It would not be wrong if some say PID feedback control is the most widely and commonly used in industrial control applications. The PID controller algorithm is sometime called **three-term control** because of the three constant parameters that define the controller.

- P Proportional
- I integral
- D Derivative

With the collective action of these three parameters, the system is controlled around the set point. However, we need to tune these parameters to give us the best results, once properly tuned PID controller algorithm, the control action control active behaves in such as way to give us desired value (Temperature, humidity, etc.)

It is not always compulsory to use P, I, and D control actions altogether, but it's recommended to specify and choose among all these controls in accordance with the system requirement and take into your design constraints. It could be P only, PI control, PD control, or PID as well. It solely depends on your design choice and systems requirements.

PID control is a combination of P, I, and D control action. By using all these controls, the P control will prevent hunting, automatic removal of offset is done by the I action, and to cope with external disturbances D action will come into play.

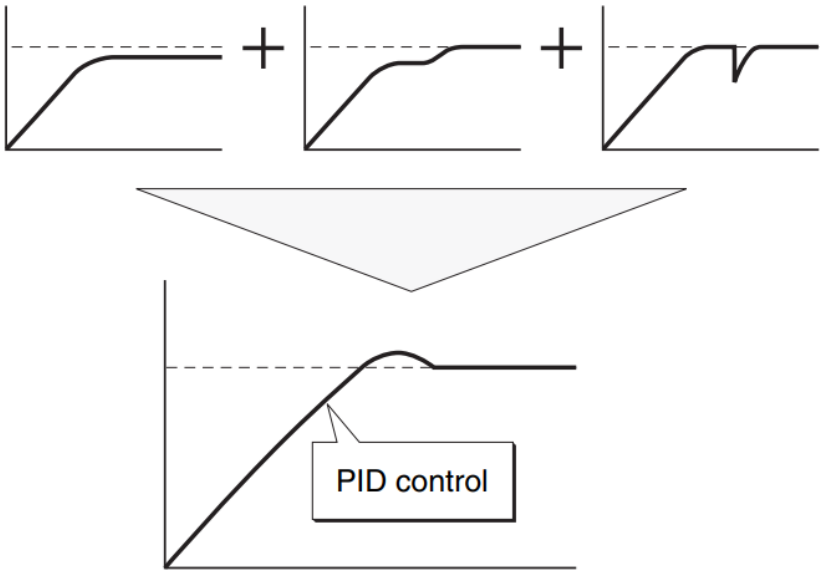


Figure 4.9 : Representation of effect of P action, I action, D action and altogether as PID

The mathematical relations

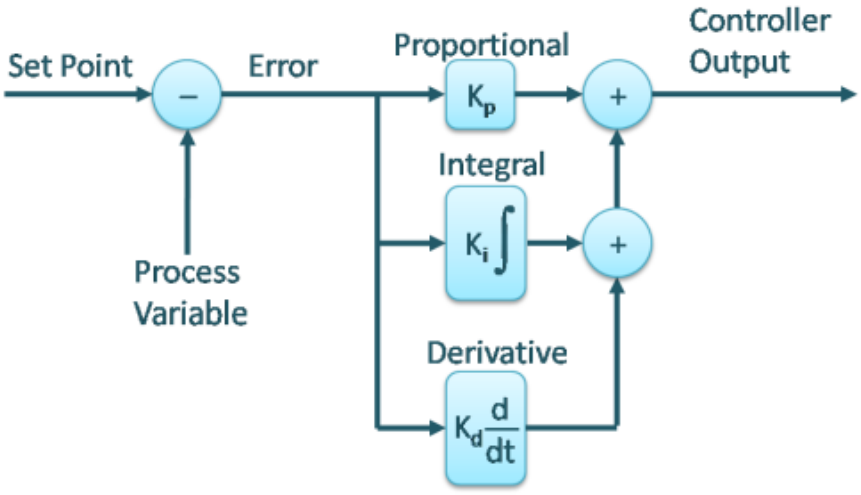


Figure 4.10: Standard PID configuration

PID general algorithm

$$u(t) = k_p\, e(t) + k_i \int_0^t e(t)dt + K_d \frac{d}{dt} e(t)$$

Proportional Term

$$P_{out} = k_p e(t)$$

Integral Term

$$I_{out} = k_i \int e\,(t)dt$$

Derivative Term

$$D_{out} = k_D \frac{d}{d(t)} e(t)$$

Were

k_p : Proportional Gain

$K_i = \frac{K_p}{T_i}$: Integral Gain

$K_d = K_p T_d$: Derivative Gain

4.2. TPI Algorithm

Two algorithms were found in literature, and both works on same principle and give the same results, however mathematical they are bit different. Algorithm 1 is bit simpler to use hence in later part we are going to use Algorithm 1 instead of algorithm 2.

4.2.1. Algorithm 1

$$\mathbf{U_{n=}U_{n-1} + K_p[e_n - e_{n-1}] + \frac{K_p T_s}{Ti} e_n}$$

Where

$U_{n=}$Old Output

U_{n-1} = New output

e_n = Old error

e_{n-1} = New error

K_p = Proportional Gain

Ti = Intergral time

T = discrete time

```
function [out_new,out_old, error_old]  = fcn(error_new,kp,ti,ts,start,out_old,error_old)



if(start == 1)
    out_new =0.5 + kp*error_new ;                                          % initialization
else
    out_new =out_old + kp*(error_new-error_old) + (kp*ts*error_new)/ti;     % main algorithm
end


error_old = error_new;
out_old = out_new;
```

Figure 4.11: TPI algorithm 1 code in MATLAB function block in Simulink

4.2.2. Algorithm 2

$$\mathbf{U_{n=}U_{n-1} + \beta_\circ\ (e_n) + \beta_1\ (e_n - e_{n-1})}$$

Where,

$$\beta_\circ = \frac{K_p(T_s + 2T_i)}{2T_i}$$

$$\beta_1 = \frac{K_p(T_s - 2T_i)}{2T_i}$$

```
function [out_new,out_old, error_old] = fcn(error_new,kp,ti,ts,start,out_old,error_old )
b0= Kp*(Ts+2*Ti)/(2*Ti);                                    %Calculation of beta0
b1= Kp*(Ts-2*Ti)/(2*Ti);                                    %Calculation of beta1

if(start == 1)
    out_new = .5 + b0*error_new;                            % initialization
else
    out_new = out_old + b0*Enew+b1* +(error_new-error_old)  % main algorithm
end

Eold = Enew;
T_e_old = T_e_new;
```

Figure 4.12: TPI algorithm 2 code in MATLAB function block in Simulink

4.3. Auto-Tuning Methods

4.3.1. Ziegler-Nichols' Process Ultimate Gain Method (Closed-Loop Method)

In the 1940's Zeigler and Nicholes presented a tuning method to calculate PID gain parameters. Ziegler-Nichols' closed loop method allows calculating ultimate gain values Kcr and the period of oscillation Pcr. This method is commonly in use because of its simplicity and better approximations of the controller. However, this method has limitations too, it cannot run in an open-loop environment.

Finding the ultimate gain value Kc, is done by finding the ultimate gain which is the gain value that gives the steady-state oscillations. That means, for instant, the I and D controller are set to zero so that the influence of the P controller can be observed Hence the ultimate gain. Another intermediate parameter needs to be found which is also important. that parameter related to P only controls that is called critical period (Pcr). The ultimate period or critical periodic s the time required to complete one full oscillation while the system is in a steady state.

Tuning Steps

- Remove integral and derivative action from control. Only P action to left in the loop
- Increase the P gain gradually so that we have uniform wave or oscillation having constant amplitude.
- Record the P gain where we are having oscillation which is our Critical gain (Kcr)
- Record the critical period (Pcr) which is the gap between two subsequent peaks it can be maxima or minima.
- After you find two parameters you can find P, I & D gain by simply putting the values in the table below

	K_c	T_i	T_d
P controller	$0.5K_{cu}$	∞	0
PI controller	$0.45K_{cu}$	$\frac{P_u}{1.2}$	0
PID controller	$0.6K_{cu}$	$\frac{P_u}{2}$	$\frac{P_u}{8} = \frac{T_i}{4}$

Table 4.1: Ziegler-Nichols' Process Ultimate Gain Method (Closed-Loop Method) tuning table

4.3.2. Ziegler-Nichols' Process Reaction Curve Method

It's another Ziegler-Nichols [9] tuning process which is also an effective one. In this tuning process, step response is recorded, in which step response is measured ymf after a step with height U in the controlled variable u. the dead-time or lag L and rate of slope R are recorded. After measuring these intermediate like the previous tuning method, the P, I, and D gain can be calculated using the table below.

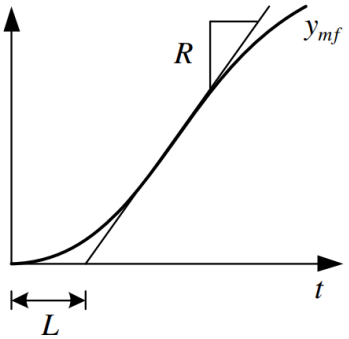


Figure 4.13: Ziegler-Nichols' open loop method: The equivalent dead-time L and rate R read off from the process step response.

	K_p	T_i	T_d
P controller	$\frac{1}{LR/U}$	∞	0
PI controller	$\frac{0.9}{LR/U}$	$3.3L$	0
PID controller	$\frac{1.2}{LR/U}$	$2L$	$0.5L = \frac{T_i}{4}$

Table 4.2: Ziegler-Nichols' open loop method tuning table

4.3.3. Skogestad's Method

Skogestad's Method tuning method also known as the Simple Internal Model Control (SIMC) tuning method, is a kind of model-based tuning method in which the function of process model parameters defined the controller parameters. It is considered to be a continuous-time transfer function. The control system transfer function T(s) is the transfer function from the set point to the process parameter.

$$T(s) = \frac{y_{mf}(s)}{y_{sf}(s)} = \frac{1}{T_c s + 1} e^{-\tau s}$$

where

- Tc is the time-constant of the control system specified by user
- τ time delay is is given the model of process
- $y_{mf}(s)$ the response of the system
- $y_{sf}(s)$ is the step by which we are geting $y_{mf}(s)$ the response of the system

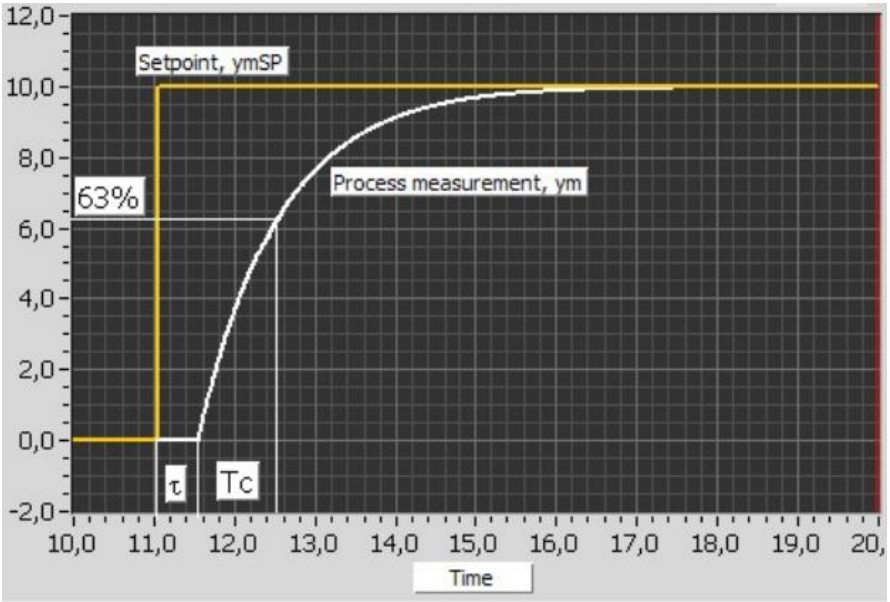


Figure 4.14: Skogestad's Method tuning method

By comparing the tracking function given by the process and tracking function given by **equation** and making some simplified assumptions regarding the delay. we can have transfer function for PID or PI only.

$$K_c = \frac{T}{K(T_c + \tau)}$$

$$T_i = \min [\, T, c(T_c + \tau)$$

Originally, Skogestad used factor c to

$$c=4$$

4.3.4. Relay-Based Tuning Method

Astrøm-Hagglund's relay-based method [13] can be considered as practical implementation of Zeigler-Nichols in simplest way. Zeigler-Nichols is bit time consuming as manually need to create uniform oscillations. This problem s solved in Relay-based Tuning method.

In this method simple ON/OFF relay controller is used in parallel with PID controller which will have tunned gains. Due to ON and OFF action, oscillation will be created automatically. After we have uniform oscillation, we can simply use Ziegler-Nichols' closed loop method to find critical gain (*kcr*) and critical period (*Pcr*).

The parameters need to find out our "High" means ON and "Low" means OFF, are U_{high} and U_{low} respectively. Once we are done with defined High and Low, we can calculate the amplitude A of the relay controller as

$$A = \frac{U_{high} - U_{low}}{2}$$

We can also set the controller signal into percentages

$$U_{high} = 100\ \%$$

and

$$U_{low} = 0\%$$

Possible solution, if the there is not relay present in the controller, can make PID controller into relay by following rules.

Kc = very large;

Ti = very large;

Td = 0

As explained before that, this method will automatically give you sustained oscillations, hence ultimate gain of relay can be calculated as follow

$$K_c = \frac{\text{Amplitude of realy output}}{\text{Amplitude of realy input}} = \frac{A_u}{A_e} = \frac{\frac{4A}{\pi}}{E} = \frac{4\,A}{\pi E}$$

Where,

$A_e = E$: is the amplitude of oscillation of error signal

$A_u = \frac{4A}{\pi}$: amplitude of the oscillation of output signal

Once you set you control on ON/OFF control it will give oscillations and K_c is calculated. Afterwards when you have oscillations, we can obtain he value of ultimate gain Pu from the signal. Finally, the gain values can be calculated using Ziegler-Nichols's table which is given below

	K_c	T_i	T_d
P controller	$0.5K_{cu}$	∞	0
PI controller	$0.45K_{cu}$	$\frac{P_u}{1.2}$	0
PID controller	$0.6K_{cu}$	$\frac{P_u}{2}$	$\frac{P_u}{8} = \frac{T_i}{4}$

Table 4.3: Ziegler-Nichols' Process Ultimate Gain Method can be used for Relay-based tuning method

4.3.5. Simulink PID Auto-Tuner App

In Simulink there is a built-in Simulink block that is designed as PID controller, where all you need to do is put your PID gains, built-in PID algorithm will work, with no need to write any kind of code in it.

Moreover, within PID Simulink block, there is a tuning App by using which you can tune your system as well.

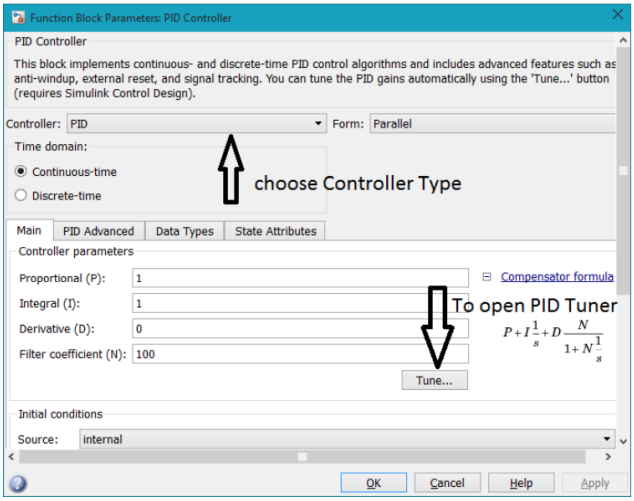


Figure 4.15: PID controller Simulink block interface

In the above figure.4-15 shown, once you click the PID block this interface will show up in which you can see there is an option to select between different types of controllers as well, like PID, PI or P only control.

Moreover, Tune button you can see is to tune your system which is attached to the model .it will calculate the optimum gain values. Furthermore, there is an option to choose from continuous-time and discrete-time as well accordance with your system/model requirement.

Example

Simulink model

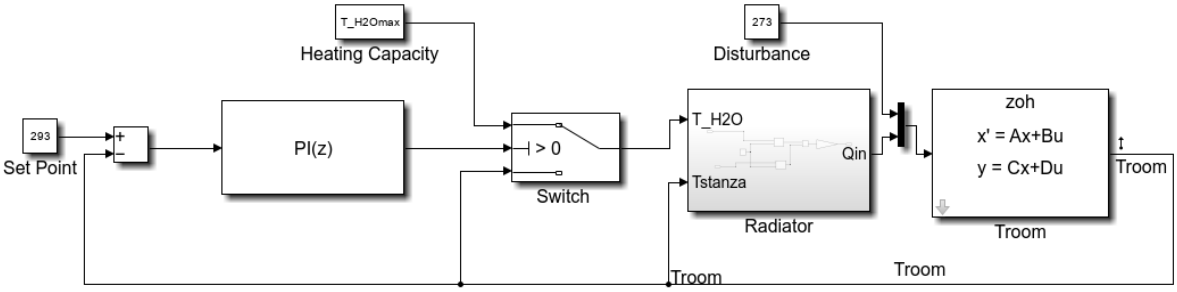


Figure 4.16: Basic Simulink model with PID Simulink block

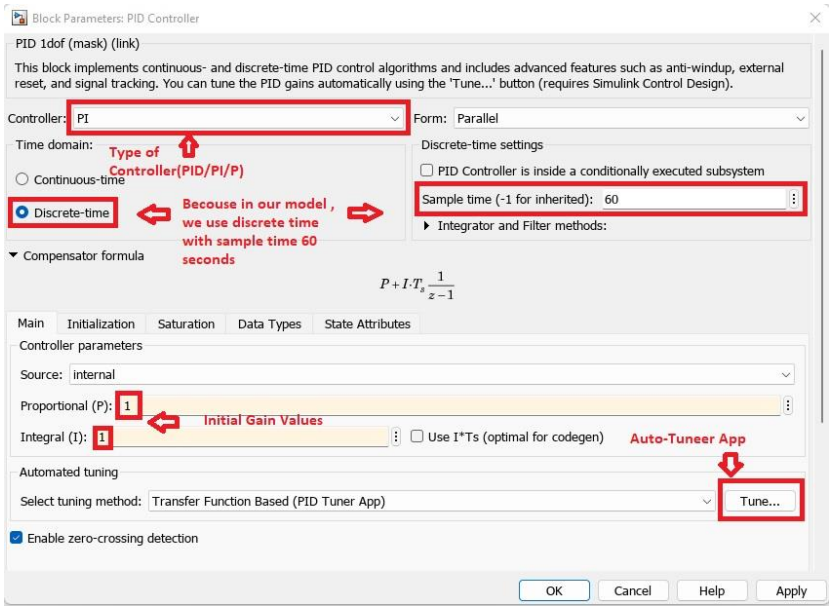


Figure 4.17: Parametrs that are important to set inside the PID Simulink block

After clicking the Tune App button, it will lead you to the PID tuner App, where it will show you the tunned response and if you are convinced with the result, it will update the tuned gain values in the PID tune block, hence model can run using tuned values.

PID Tuner toolbox

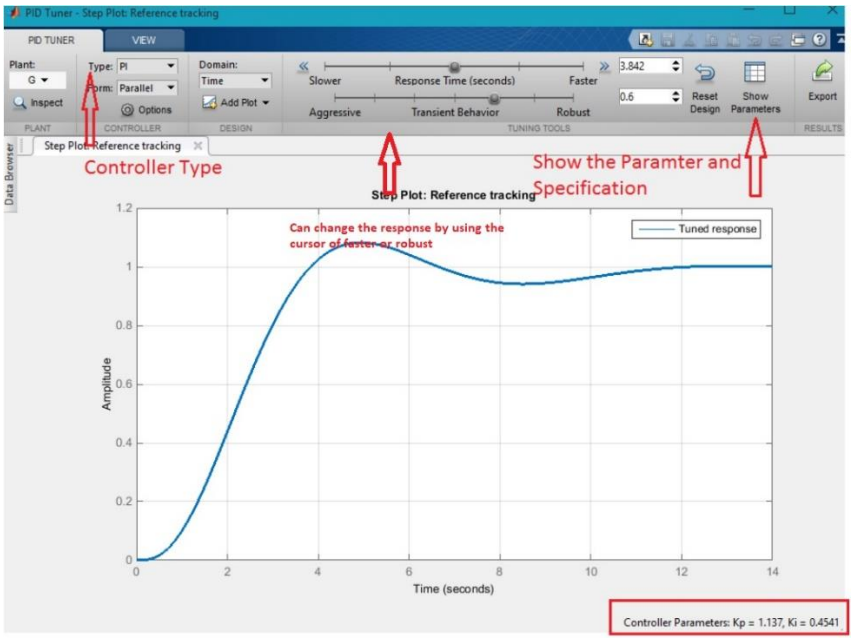


Figure 4.18: Interface of Simulink PID auto-tuning App

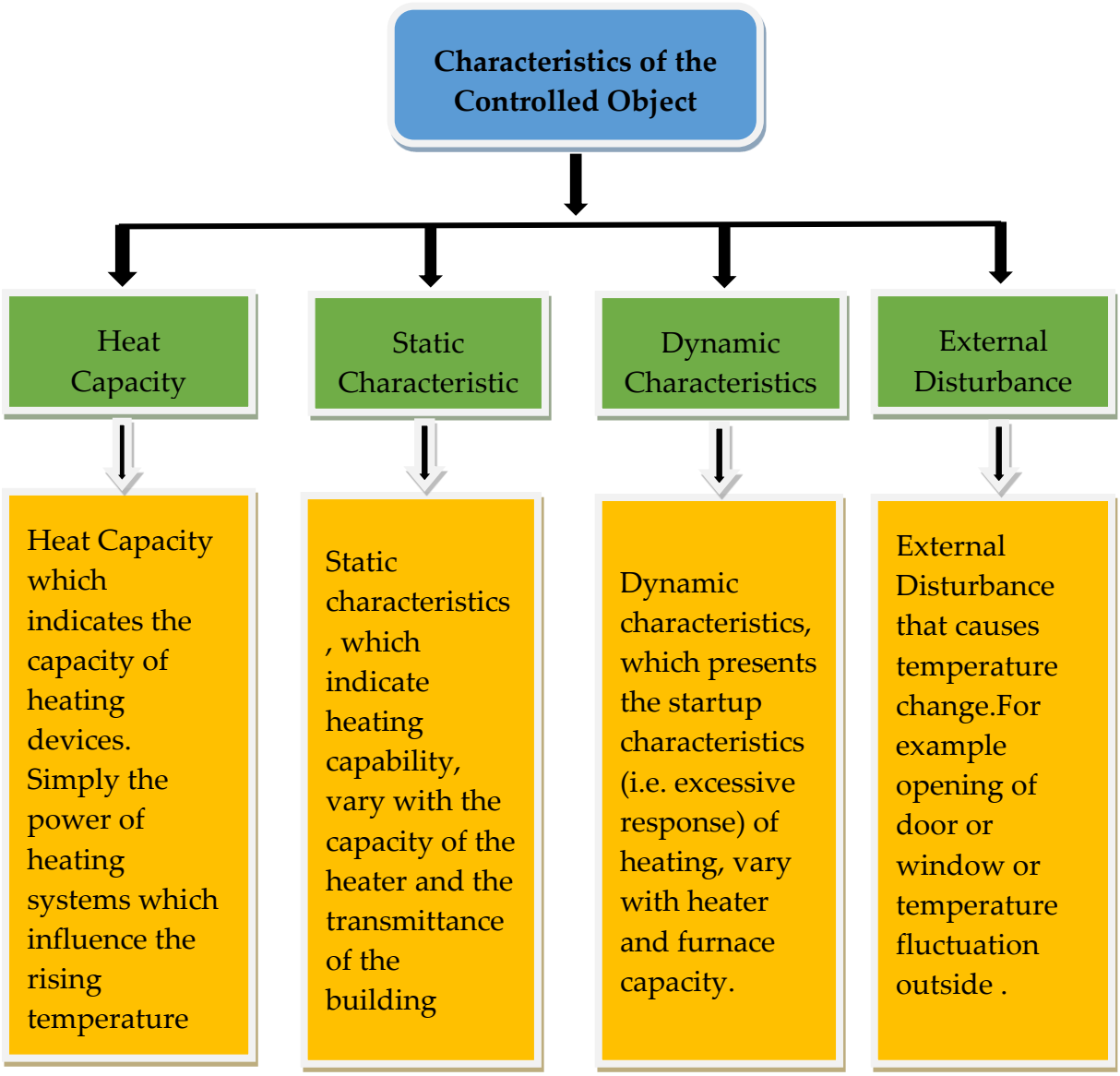
This is the interface of the auto-tuner app that shows different selections where we can change the gain values as we desire. The tuner app gives us the gain values contrast between the robustness and fast as well. However, you can change the robustness or if you want a fast response, it's possible by simply using the curser as pointed out in the above picture. After satisfaction with the response, there is an update button in the app, by clicking that tuned gains will be updated in the PID block in the Simulink model.

4.4. Characteristics of the Controlled Object/System

Before designing our control system, the most important part is to learn the characteristics of the systems we are trying to control.

4.4.1. Characteristics of the Controlled Object/System

Best way to understand Characteristics of the Controlled Object/System is identify all the parameters that influence Our systems and try to understand of each parameter.



4.4.2. Heating Capacity

Heating capacity is simply the capacity of the systems or power of heating of the equipment. It depends on the heating equipment and the rating it is operating. Let's take an example of the three-heating equipment namely to fan coil, radiator, and convertor. Among all these heating equipment fan coils is faster than radiator and convertor. As you can see in figure 4.19. The temperature gradient is high for the fan coil compared to the radiator and convertor. The response time is simulated in DesignBuilder software in order to compare how long it takes for each piece of equipment to reach the set point which is 22 °C. Note that this simulation is carried out keeping all the influential parameters especially energy class kept the same for each case. This particular simulation is done by choosing energy class A1.

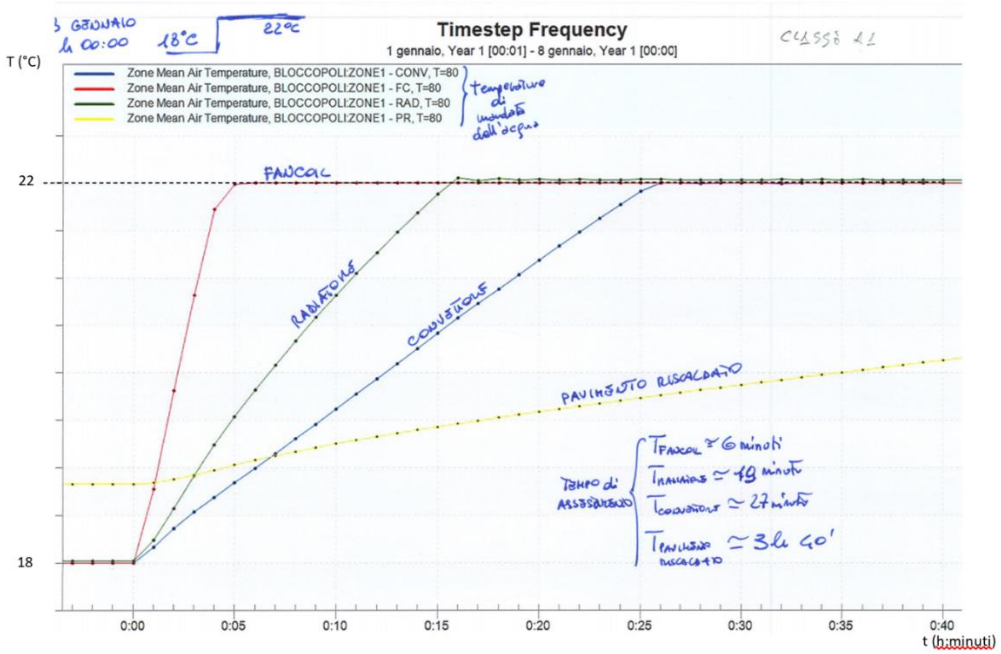


Figure 4.19: Response times fan coil, radiator, convector and heated floor in class A1

Fan Coil	Radiator
50 °C	50 °C
60 °C	60 °C
70 °C	70 °C
80 °C	75 °C

Table 4.4: The practical delivery temperature of radiator & fan coil

The heat capacity in our model is defined by delivery temperature of the equipment in above table the range of delivery temperature is defined for radiator and fan coil as well.

4.4.3. Static Characteristics

In previous chapters, we designed a model that represents the Room/Building zone in which there is an important parameter that plays in important part in defining the Characteristics of the room. That parameter is the transmittance (U) of the room. Moreover, on base of that transmittance (U) can also differentiate the energy class as well. The transmittance is considered as static characteristics of the system as it does not change.

4.4.3.1. Energy Class

Energy class is defined by the transmittance of the walls. In simple words, its represents that how much heat can pass from the walls, if the transmittance is low that means insulation of the wall is not that good falling into low energy class categories and that also means heat losses are expected to be more.

Moreover, if the transmittance of the walls is less then our system is experiencing more load and hence in order to fulfil heat demand, our system should be turned on more often. In order words, if transmittance is high that means energy class is high , a well-insulated room , heat demand is surely be less.

The transmittance of the walls therefore remains the only option, it presents a continuous variable that theoretically has no limits, although to remain as realistic as possible you will avoid too high or low values of this variable. Moreover, since EPgn is quite sensitive to changes in transmittance, vast values for energy can only be obtained in the 0.2-0.65 domain; increasing this range (to 0.1-1 which remains relatively realistic for buildings) will result in a satisfactory codomain to be able to describe all classes.

Below is a table of transmittance values for each energy class and for the different types of terminals. These values of transmittance are really essential to define energy class for each of the system.

Energy Class	**Fan Coil**	**Radiator**
A4	0.051	0.090
A3	0.088	0.128
A2	0.151	0.183
A1	0.214	0.239
B	0.278	0.295
C	0.358	0.365
D	0.488	0.478
F	0.927	0.853
G	1.192	1.084

Table 4.5: Transmittance values for each energy class

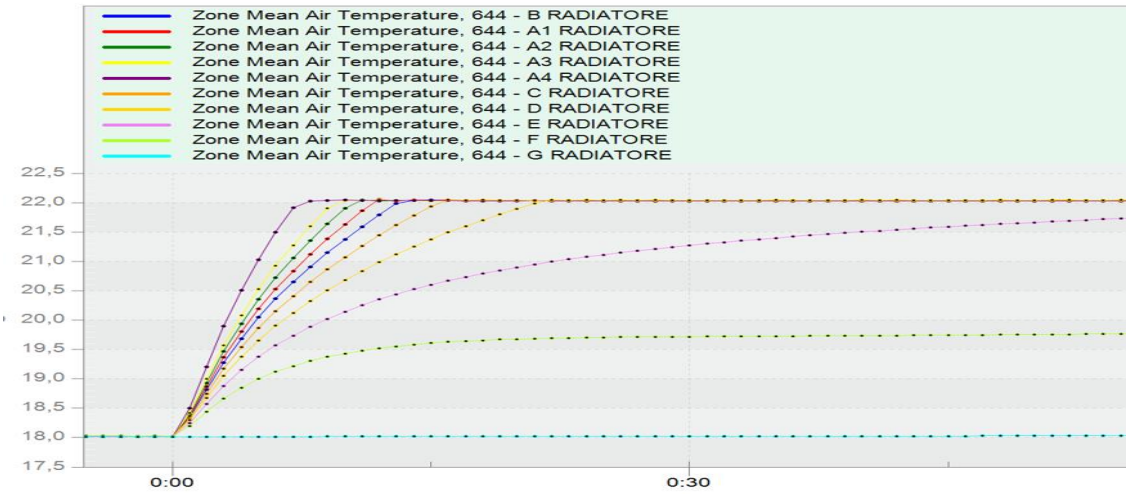


Figure 4.20: Response time of for each energy class simulated in DesignBuilder

The above figure.4-20 shows that response time for radiator how much time it took for radiator to reach set point. In this example, everything is kept constant, radiator delivery temperature, external temperature, room dimension only the energy class is varied to see the response of each energy class. As you can see from graph, that for A4 energy class for system to reach set pot which is 22 °C is less that means A4 energy class is more insulated and heat losses are less.

On the other side, energy class F is the slowest among all other as it looks longer time to reach set point. This also show that to response time to reach set point, not only depends on the heating system but also the energy class and the extent of disturbances. Logically, the lower energy classes have higher settling times depending on the terminal inserted: for the fan coil the differences are minimal, ~minute; while for the radiant surfaces substantial differences can be noted, hours.

4.4.4. External Disturbance

External disturbance is due to multiple factors, the most common one is the temperature variation outside of the room or building. The other factors could be opening of window, door or you can say presence of person in room. All these factors influence the control system, hence are important to consider all these factors.

Factors

- Outside temperature
- Open door
- Open window
- Presence of other heating devices
- Presence of person in room

4.4.5. Case Study

Let's divide the case in such a way that in first case we are keeping the room and external parameters constant only varying the heating capacity of the heating system and in second case we will change the room and external disturbance parameters and keeping the heating capacity and heating equipment same.

In order to compare the response, simple model below is designed, in which heating device is turned ON always for specific period of time until which we can cross our set point to observe the influence of parameter we are taking into account.

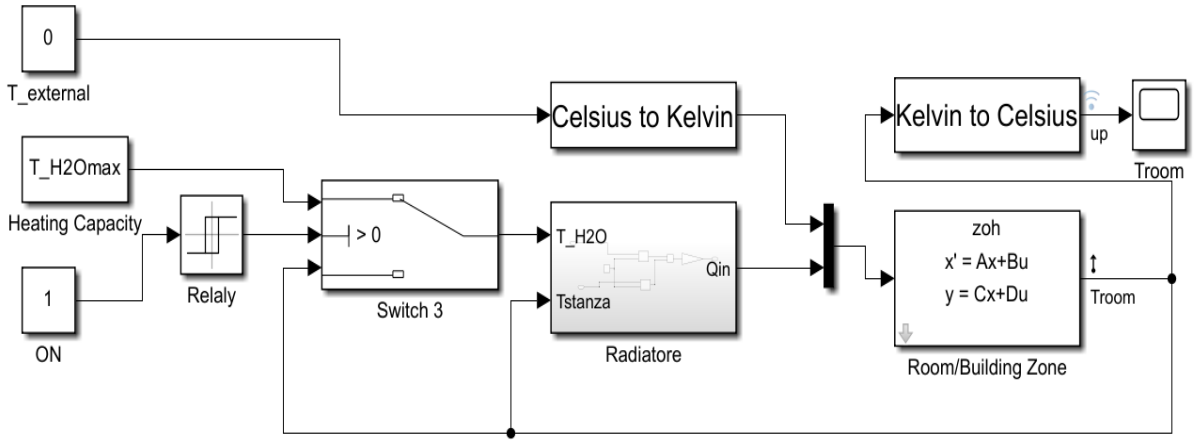


Figure 4.21: Simulink model to understand the behaviour of the system under change in influential parameters

4.4.5.1. Case Study 1 (Changing Heating Capacity)

In this case we want to observe the impact of capacity of heating system keeping all the other parameters constant.

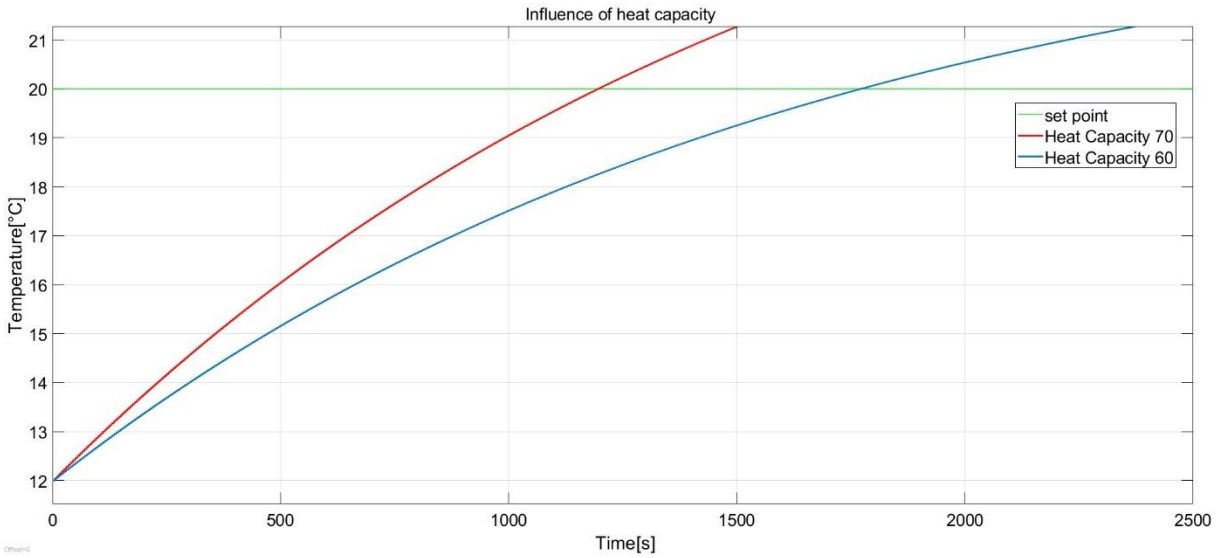


Figure 4.22: Temperature response by changing the heat capacity and its effect on time taken to reach set point

Parameters	Case 1 (Red)	Case 1 (Green)	Comment
Heating Equipment	Radiator	Radiator	No Change
Energy class	B	B	No Change
Transmittance of energy class	0.295	0.295	No Change
Set point	20 °C	20 °C	No Change
Start temperature	12 °C	12 °C	No Change
External disturbance / T external	0 °C	0 °C	No Change
Heating capacity	**70 °C**	**60 °C**	**Change**
Time taken to reach set point	**1190 s**	**2360s**	**Change**

Table 4.6: Changing the heat capacity and its effect on time taken to reach set point

4.4.5.2. Case Study 2 (Changing Energy Class)

In this case study, changing energy class also show as different response and better energy class approaches set point quickly which is A1 comparing to B.

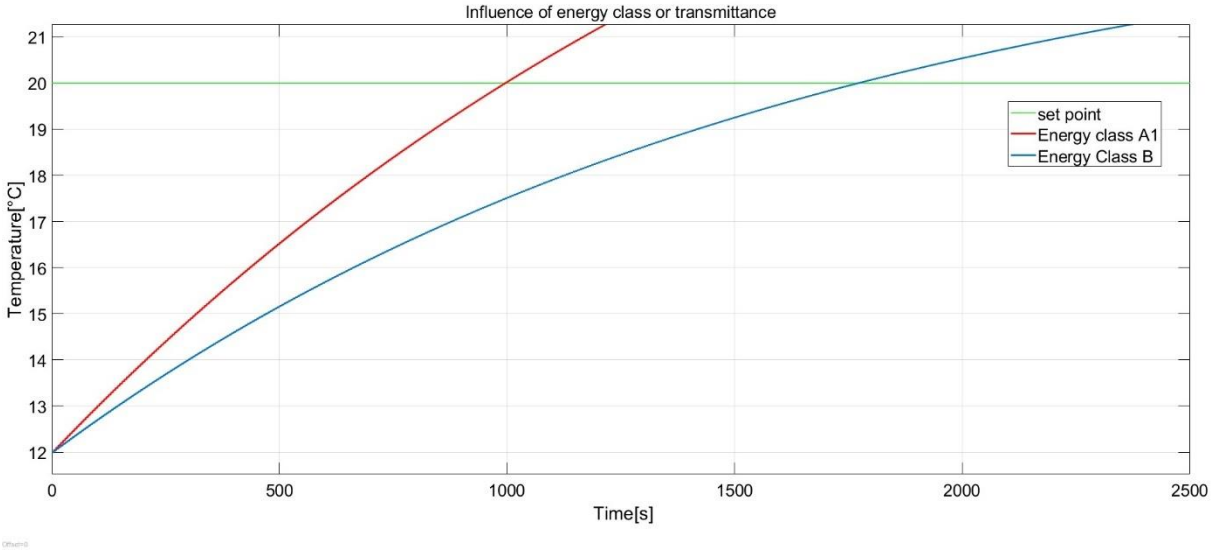


Figure 4.23: Temperature response by changing the Energy class/Transmittance and its effect on time taken to reach set point

Parameters	Case 1 (Red)	Case 2 (Green)	Comment
Heating Equipment	Radiator	Radiator	No Change
Energy class	**A1**	**B**	**Change**
Transmittance of energy class	**0.239**	**0.295**	**Change**
Set point	20 °C	20 °C	No Change
Start temperature	12 °C	12 °C	No Change
External disturbance / T external	0 °C	0 °C	No Change
Heating capacity	70 °C	70 °C	No Change
Time taken to reach set point	**995 s**	**1200 s**	**Change**

Table 4.7: Changing the energy class and its effect on time taken to reach set point

4.4.5.3. Case Study 3 (Changing Disturbance)

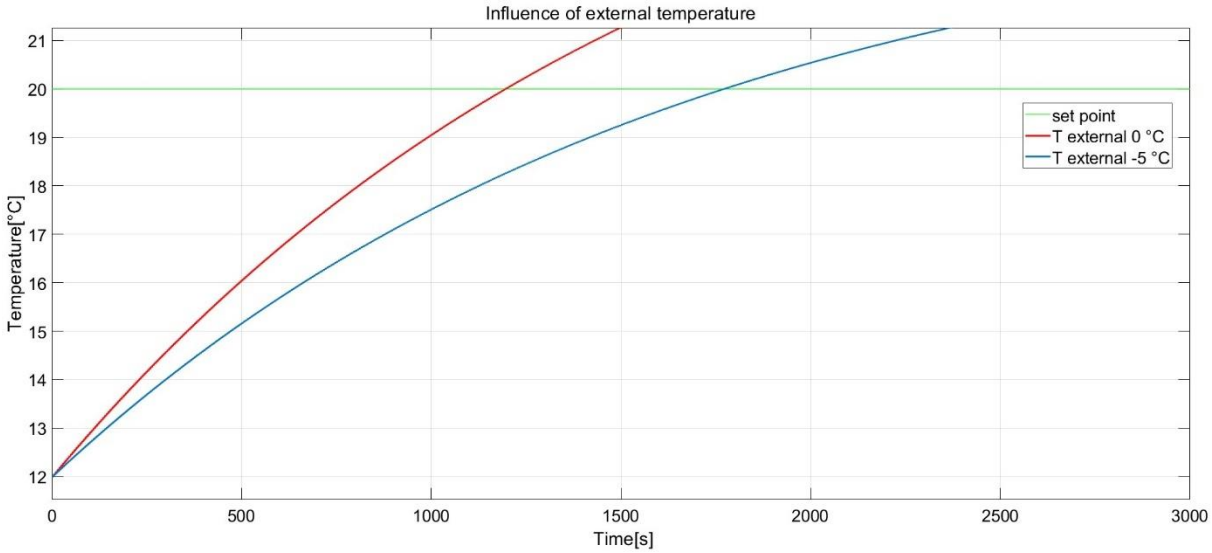


Figure 4.24: Temperature response by changing the heat capacity and its effect on time taken to reach set point

Parameters	Case 1 (Red)	Case 1 (Green)	Comment
Heating equipment	Radiator	Radiator	No Change
Energy class	B	B	No Change
Transmittance of energy class	0.295	0.295	No Change
Set point	20 °C	20 °C	No Change
Start temperature	12 °C	12 °C	No Change
External disturbance / T external	**0**	**-5 °C**	**Change**
Heating capacity	70 °C	70 °C	No Change
Time taken to reach set point	**1200 s**	**1774 s**	**Change**

Table 4.8: Changing the external temperature and its effect on time taken to reach set point

5 Development of Control Design

Until now, we have the basic model in Simulink, the parameters that influence our system, how system behave by changing the influence parameters, multiple methods of tuning. In this chapter, will try to implement all the knowledge we gathered from previous chapter and try to get to the final phase of TPI temperature control algorithm.

Another thing that is worth mentioning is that as per company's requirement, only PI controller is to be designer rather than three term controller which is PID due to some constraints involved with derivative gains that will be discussed in later part of the chapter.

There will be two approaches we will implement and conclude which path is best to follow and why with practical reasoning. Each approach is different from each other, by how we tune the system.

- Stationary system (Tune with respect to each energy class)
- Dynamic system

5.1. Stationary System

In this approach, we will try to tune the system on bases of energy class. As in the previous chapter we discussed what are energy class, what parameters defines the energy class and which parameters in out Simulink model defines the respective energy class. The transmittance (U) is the parameter that defines the energy class and we have calculated the Transmittance(U) for each energy class using DesignBuilder.

With respect to each energy class will tune the controller and get the respective gain values by using Simulink Auto-tuner App for radiator and fan coil as well. Once we have calculated all the controller's gain that can be recorded and whenever user define the energy class in the thermostat the respective gain values will be feed into the temperature control algorithm.

TPI Algorithm 1

$$\boldsymbol{U_{n=}U_{n-1} + K_p[e_n - e_{n-1}] + \frac{K_p T_s}{Ti} e_n} \quad 5.1$$

Where,

$U_{n=}$Old Output

U_{n-1} = New output

e_n = Old error

e_{n-1} = New error

K_p = Proportional Gain

Ti = Intergral time

T = discrete time

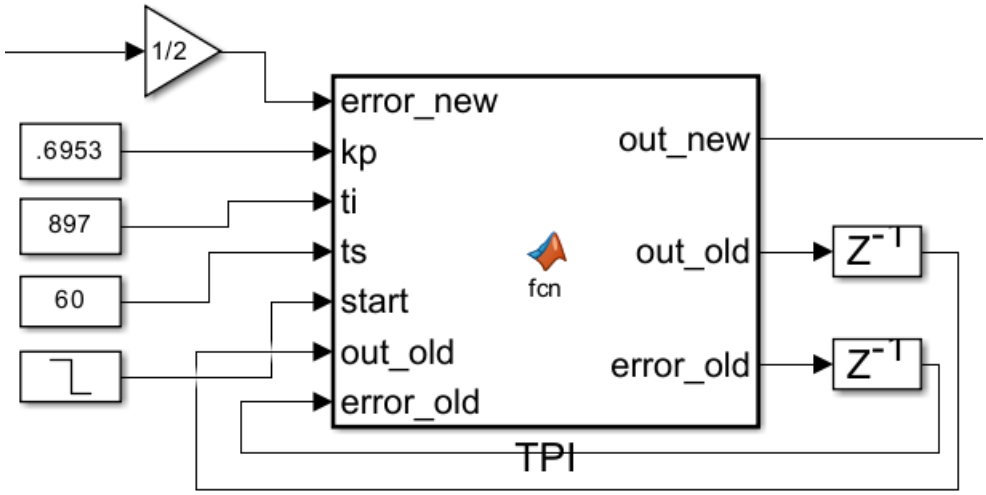


Figure 5.1: MATLAB function block for TPI algorithm

In this algorithm the most important variable that needed to be inserted are proportional gain Kp and integral time Ti that needed to be inserted energy time energy class changes.

Let's using PID tuner App in Simulink to tune the system for each energy class (w.r.t transmittance values)

System parameters

- Set Point: 20 °C
- T max (Heat capacity): 60 °C (Discharge temperature of the radiator)
- T External: 0 °C (Outside Temperature or disturbance)

After setting all the model parameters, using PID tuner App came up with proportional (Kp) and integral gain (Ti) values for each energy class. How is it is done is simple, by changing the transmittance value as shown in the table for each energy class and tuned respectively. The tuned values of proportional (Kp) and integral gain (Ti) are shown in the table

Energy Class	Transmittance (U)	Kp	Ti
A4	0.090	0.1800	495.0
A3	0.128	0.2913	593.9
A2	0.183	0.5069	775.6
A1	0.239	0.6953	897.0
B	0.295	0.960	1052
C	0.365	1.164	1249
D	0.478	-	-
F	0.853	-	-

Table 5.1: Tuned values of Kp & Ti for energy class using Simulink auto-tuning App

This table is feed into our microcontroller in the thermostat, and it gives the values to controller whenever user changes the energy class.

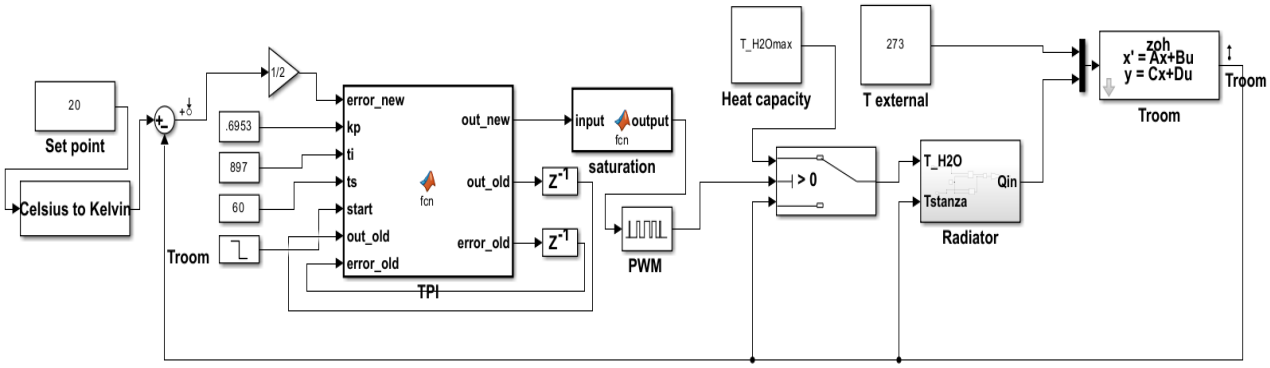


Figure 5.2: Simulink model used to validate the tuning with respect to energy class approach

5.1.1. Case study

Let's have a case study to know this approach is working or not if it's how effective this approach is. The only difference between case 1 and case 2 has is difference in maximum delivery temperature of the radiator (heating capacity) of the system.

For both cases energy class(A2) is same, moreover have same Kp & Ti values, hence if our approach of tuning according to energy class (U) is true both should have given is same results, regardless of different heating capacity. (Tmax)

Parameters	Case 1 (Yellow)	Case 2 (Blue)	Comment
Heating Equipment	Radiator	Radiator	No Change
Energy class	A2	A2	No Change
Transmittance of energy class A (U)	0.239	0.239	No Change
Set point	20 °C	20 °C	No Change
Start temperature	12 °C	12 °C	No Change
External disturbance / T external	0 °C	0 °C	No Change
Heating capacity	**60 °C**	**70 °C**	**Change**
Proportional gain (Kp)	**0.6953**	**0.6953**	**No Change**
integral gain (Ti)	**897**	**897**	**No Change**

Table 5.2: Case study to validate the tuning with respect to energy class approach

5.1.1.1. Results of case 1 and case 2

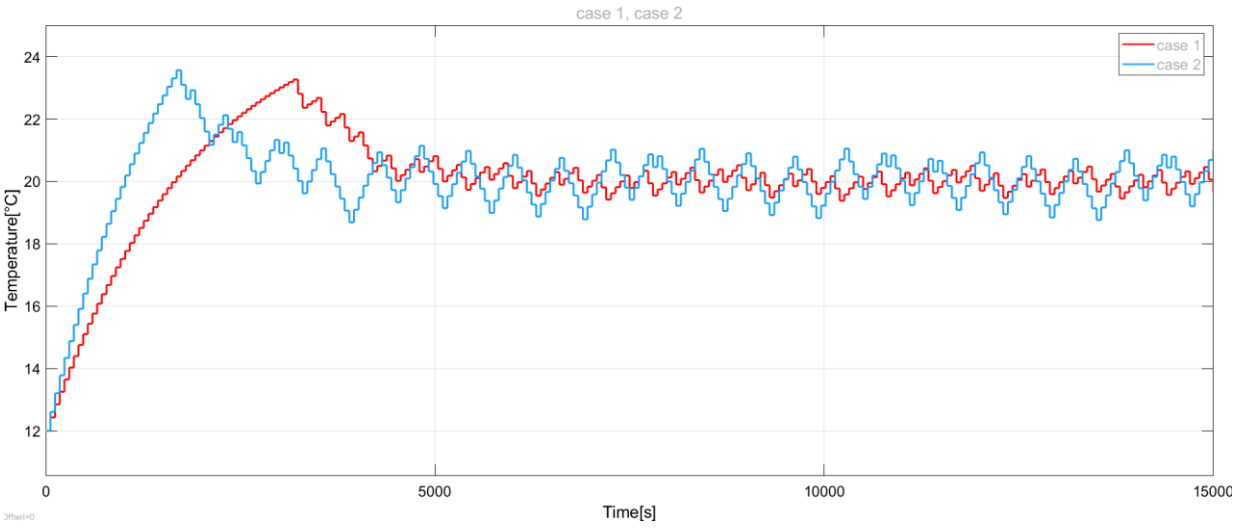


Figure 5.3: Result of Cas 1 & case 2

In this figure 5.3 can see that case 1 giving us better response then case 2 that is exactly because for case 1 we have tuned values and for case 2 that tuned values of case 1 are not suitable for case 2. However, we did tune on base of energy class A2, so both should have given us same result but that is not the case.

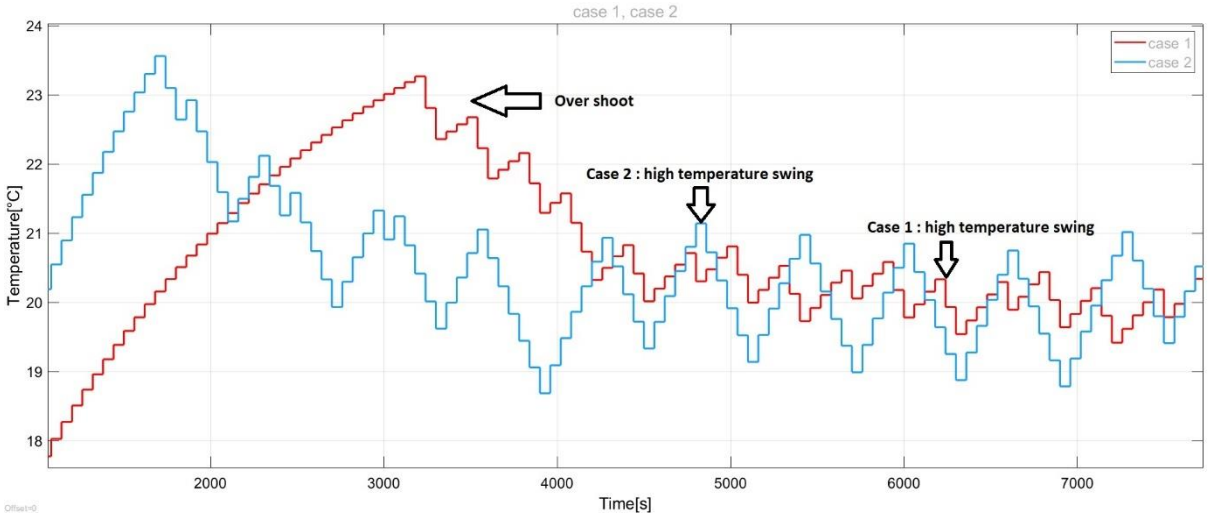


Figure 5.4: Result comparison of case 1 and case 2

From above figure 5.4 you can see that in both case we experience high overshoot, that will be solved in late part and solution to cope with overshooting will be explained. On the other hand, is worth noting that in case 2 temperature differential is higher as compared to case 1. The temperature differential for case 1 is around ± 0.6 °C and for case 2 it is ± 1.0 °C around the set point.

5.1.2. Comment & conclusion of this approach

5.1.2.1. Comments

- First and most important constraint in this approach is, in this approach user need to define/insert the energy class first in the thermostat while setting up the device. If user don't know the energy class how user can define the energy class. Moreover, what if the room where user want to install the thermostat is in between energy class A2 and A2. What if the room is really fall in energy class A1 and have modification in it.
- As this method tune the controller with respect to energy class or transmittance value (U). As discussed in the previous chapter that are the parameters that influence are system, in which we concluded that not only energy class that define the system but other factor like external disturbance, heating capacity etc.
- As seen in the case study, we saw that with different heating capacity or if we use two different heating systems which have different power, however using same proportional gain (Kp) & integral gains (Ti). The results were different, the system was tuned for one and was not for other.
- Same could be seen if heating capacity and energy class are same, Kp and Ti are same, if outside temperature is different, then again, the results show consistency.

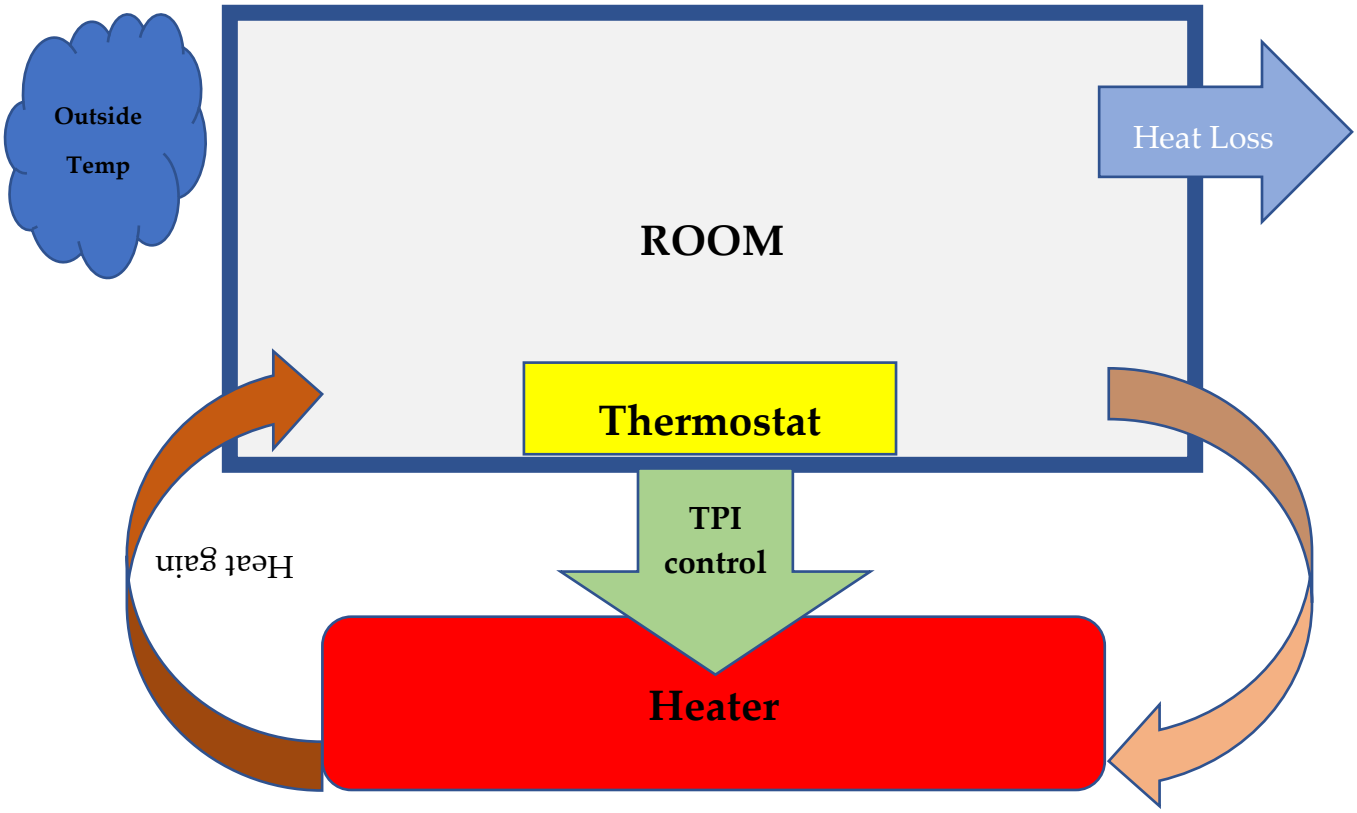
- At last, even everything same and if we change set point for example 20 and 22 respectively. Again, the result will show some sign of insistency.

5.1.2.2. Conclusion

In this approach of tuning with respect to energy class have lot of constrains as mentioned above. So, these are valid facts that questions the working of this approach. Therefore, we have developed a dynamic controlling system which will detect the heat again and heat loss (load of system). And on base of that, it will the calculate the gain values. It will tune once the device turned on or wake up and whenever we change the set point, it will go through first tuning and inserting Kp and Ti dynamically according to the nature of the system.

5.2. Dynamic control system

Due to inconvenience results and some important constraints involved in previous method, company wanted to develop a dynamic system that can tune itself according to heat needed by the system. This tuning method will take into account all the parameters in account like eternal disturbance, energy class (transmittance), and heating capacity of heating system as well.



As explained in previous chapter to extent, the characteristics of our system, how our system behave on influence of governing parameters. In previous chapter we learn,

how system changes once we change the parameters. In order to re-cap, perhaps it would be good idea to conclude those parameter's influence here as well briefly.

The parameters in the model on which our control system depends on are

- Transmittance (U) / Energy class
- Heating capacity of heating device (Radiator or fan coil)
- External disturbance (Outside temperature)
- Others which have minor influence

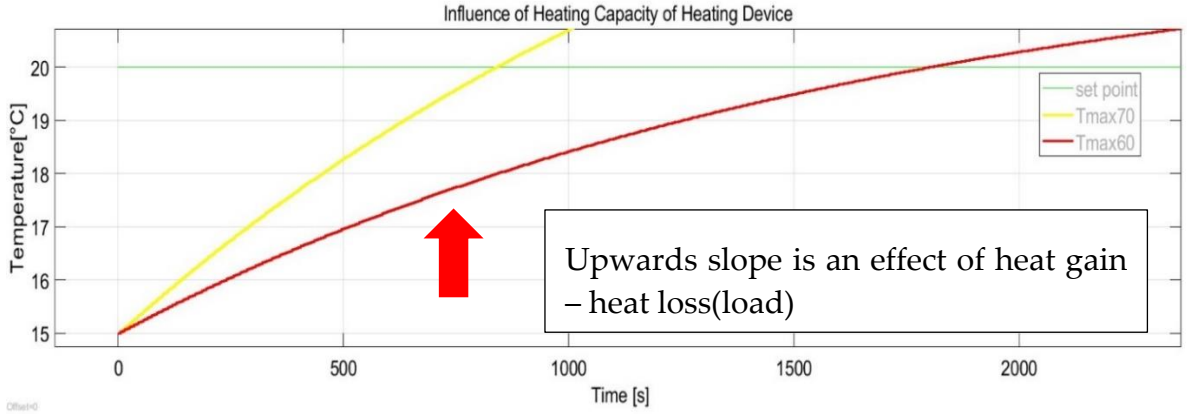


Figure 5.5: Temperature response that show that temperature gradient depends on net effect of heat capacity and system load (transmittance & external temperature)

In the above temperature response, only difference is that once the delivery temperature of radiator is 70 °C (Yellow) and once is 60 °C (Red), other variables are kept constant. The slope is different in both case, yellow line as more slope than red line. From here we can make a conclusion that upwards slope depends on is the result of heat gain minus heat loss.

Upwards slope = Heat Gain – Heat loss (Load)

Upwards slope = Heat Capacity – Transmittance – External temperature

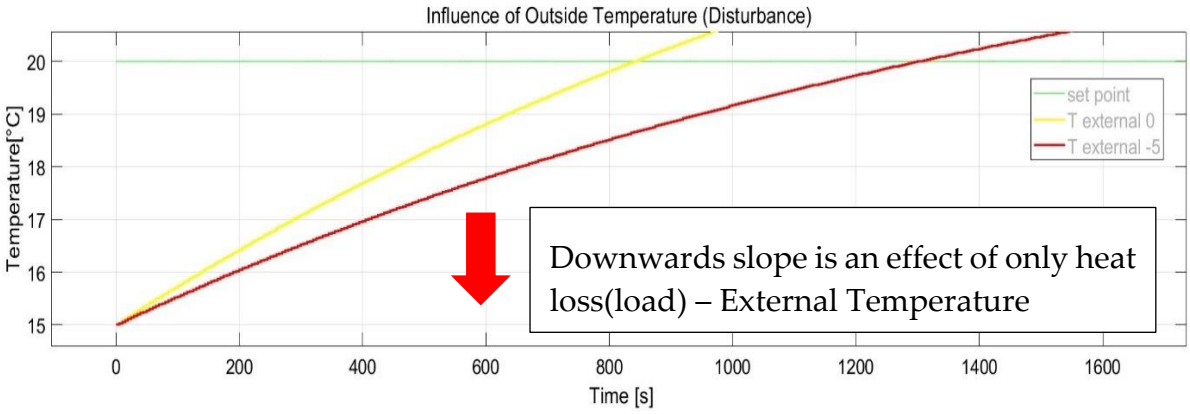


Figure 5.6: Temperature response that show that falling temperature gradient depends on external temperature

In the above graph you can see that all the other parameters are same but when external temperature is – 5 °C (Red) the response is bit slow that when external temperature is 0 °C (Yellow).

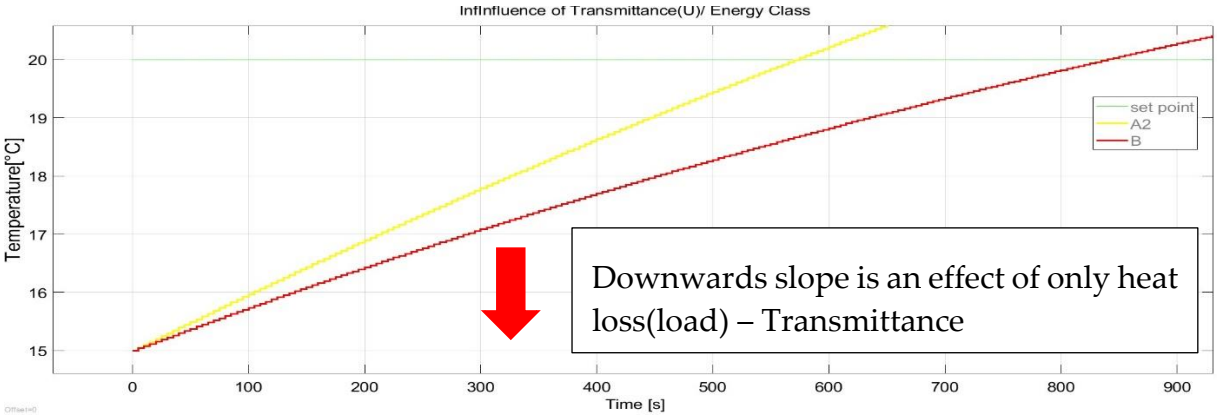


Figure 5.7: Temperature response that show that falling temperature gradient depends on Transmittance

Likewise change in external temperature, changing transmittance will have same behaviour. Moving from energy class A2 to B means lowering the transmittance values, hence heat losses will be more. That's why the slow for B (Red) is lower than A2 (Yellow)

Therefore, conclusion can be made that downwards slope depends on load on system and two parameters that influence the load which is transmittance and external temperature or outside temperature.

Downwards slope = Heat loss (Load)

Downwards slope = Transmittance + External temperature

In the light of above discussion, it is worth noting that upwards slope or can say temperature gradient is important to know because it counts for heat gain minus heat loss. In other words, it is the common effect of all the variables that define our system.

5.2.1. Tuning Method

As in previous chapter, multiple tuning methods were discussed, our approach is inspired by Relay-Based Tuning Method and Ziegler-Nichols' Process Reaction Curve Method. In our approach of self-tuning, the temperature gradient or upwards slope will be calculated across two specific point (temperature) around set point, the temperature change with respect to the time it took will tell us the temperature gradient or net rate of heat gain to the system. If we tune the system with respect to the net rate of heat gain or temperature gradient or upwards slope, regardless of the system parameters, like transmittance, external temperature, or heat capacity, if the system is giving that particular slope and we have tuned values for that slope. the result will be same, doesn't matter the properties of the system or heating device.

5.2.1.1. Proportional Band

Here is the point where, the concept of proportional band should be clear before we go any further. The proportional band is the parameter in which its set's the range for the proportional action to be performed. If the process variable in our case it is temperature is within the proportional band, the output of the controller is proportional to the deviation of temperature from the set point. whenever the temperature is with the proportional band it will vary from 0% to 100% percentage according to the error of the system.

However, if the temperature is outside of the proportional band, the output of controller could be 0% or 100 %, 0 means heating system is fully ON if output is 100 % and OFF if output is 0% respectively. If the temperature is lower, then the proportional band it output will be 100 % and if the temperature is above the proportional band, it will be 0 for whole cycle.

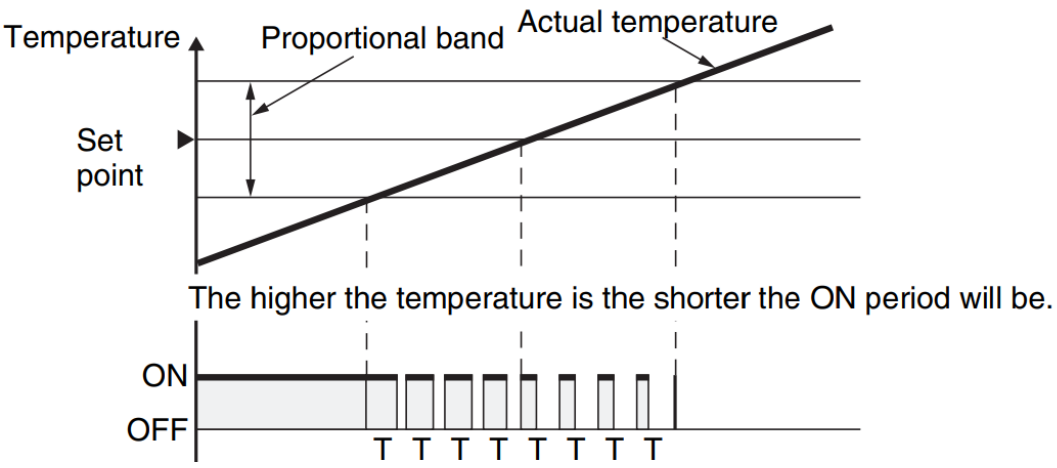


Figure 5.8: Concept of proportional band

The set point is the in the middle of the proportional band, if set point is 20 °C and proportional band is 2 °C then the upper limit of proportional band will be 21 °C and 19 °C will be the lower limit. As explained in previous chapter, the proportional band should have specific limit, and it depends on the heating system. As recommended in the literature the suitable value for radiator and fan coil is 1.5- 2 °C. Therefore, in our case as per company requirement, 2 °C of proportional band is considered for convenient results.

5.2.2. Approach of Self-Tuning

As our approach of tuning is inspired by Relay-Based Tuning method in which ON/OFF relay and TPI are connected in parallel and a logic switch will decide when to switch between two states(ON/OFF and TPI).In simple words, when the system wakes up , first controller will undergo ON/OFF state along which tuning will be done and once tuning is complete, it will switch to next state which is TPI. TPI will work

with tuned values to perform control action in best way possible. The temperature control configuration is displayed in fig below.

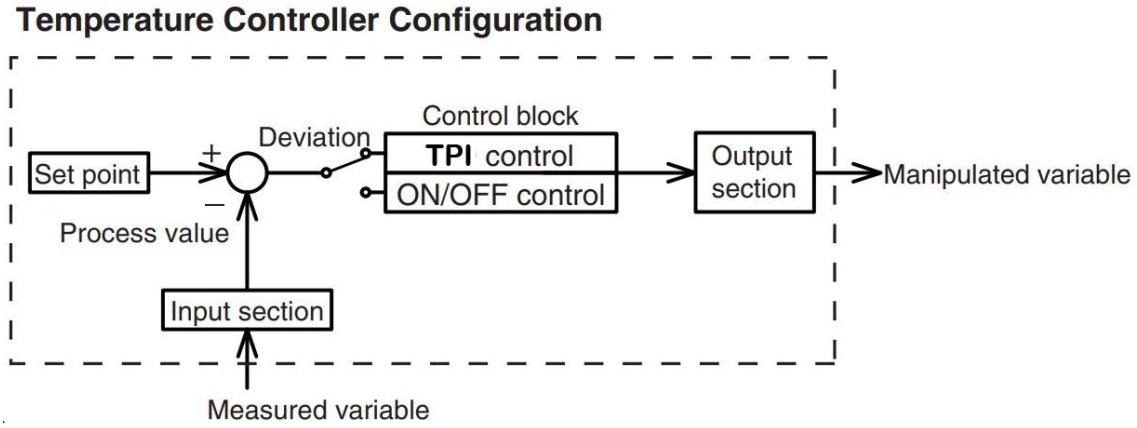


Figure 5.9: Temperature controller configuration using TPI/PI and ON/OFF controller in parallel

As shown in the fig below the temperature response we want is first ON/OFF cycle and switch to TPI. The desire is to have at least on ON/OFF cycle so that we can measure the rising slope/ upwards slope that will gives the idea of heat demand of the system. Once the rising slope is calculated, we will have respective Proportional gain (Kp) and integral time (Ti) tune values. How we will have tuned values with respect to slope will be explained later part of this chapter.

Once we have rising slope, we will feed the Proportional gain (Kp) and integral time (Ti) into TPI control algorithm which is connected in parallel to ON/OFF relay.

As you can see in the figure below once ON/OFF cycle is complete, the controller is switched to TPI.

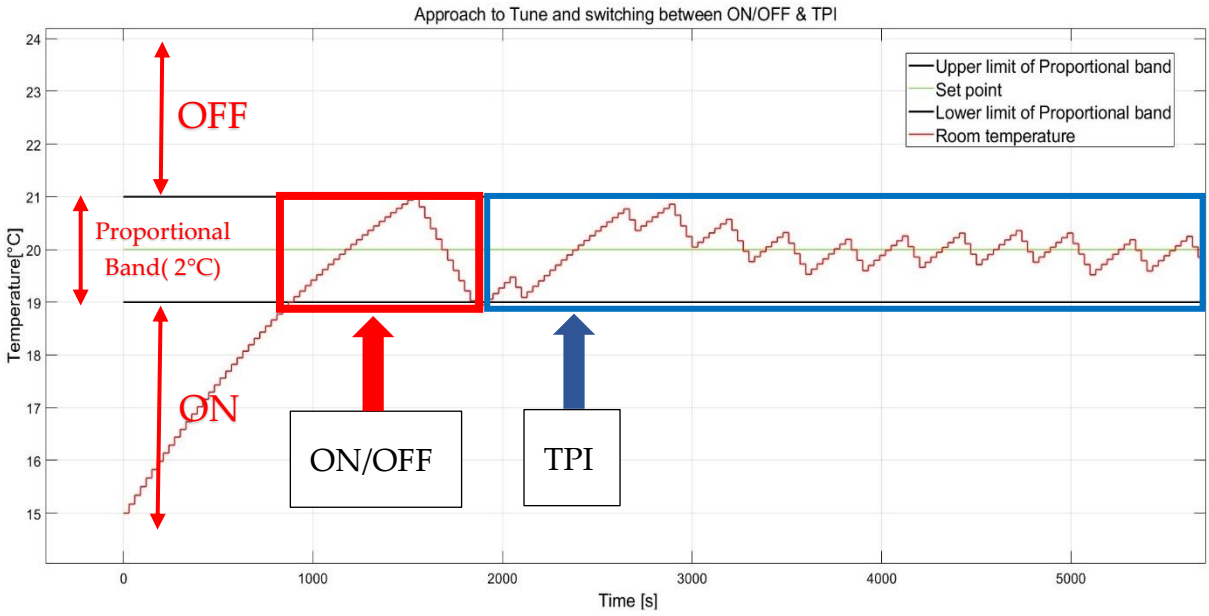


Figure 5.10: Working principle of ON/OFF & TPI/PI controller in parallel

5.2.2.1. How to Calculate the Slope

As proportional band of 2 °C degree is recommended for our system and ON/OFF relay made to operate between upper limit and lower limit of proportional band, we need to calculate slope of first rising response as shown in figure xx below, hence our controller show start counting time when it reaches lower limit of proportional band (e.g. 19 °C) and final time should be once temperature reaches the upper limit of proportional band (eg. 20 °C). let's say T1 and t2 are the temperature and time at starting point and T1 and t2 are at the end time respectively.

$$\text{Rising slope or net heat gain or temperaure gradiant} = \frac{Proportional\ band}{time\ from\ lower\ to\ upper\ limit\ of\ proportional\ band} = \frac{(T_2 - T_1)}{(t_2 - t_1)}$$

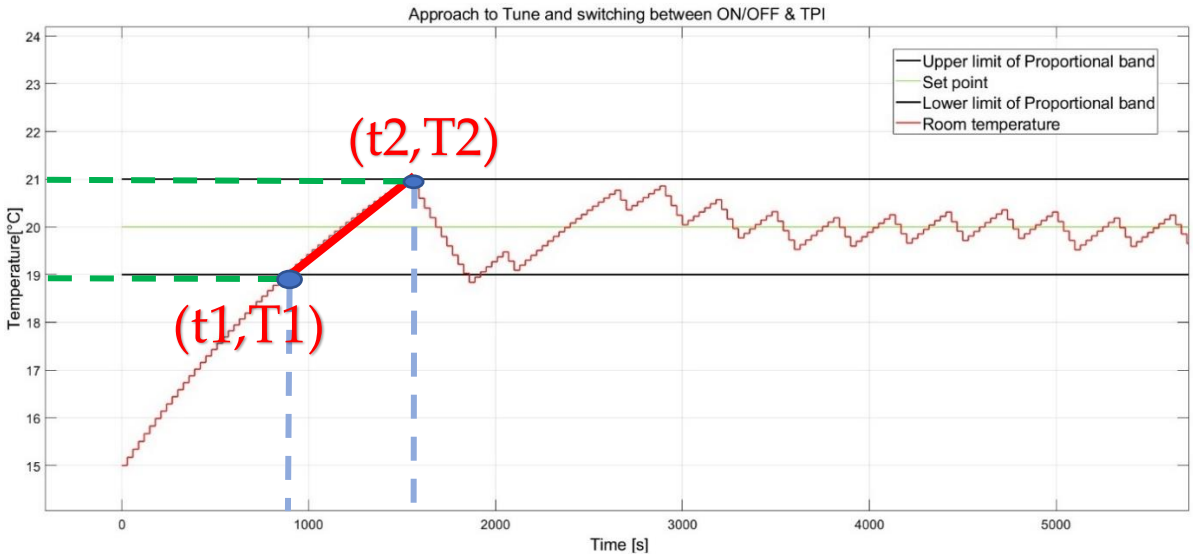


Figure 5.11: Temperature gradient/slope calculation working

In order to tune the system according to its slope or net heat gain, needed to develop a tuning algorithm that will give, optimum Proportional gain (Kp) and integral time (Ti)

5.2.3. Development of Generic Tuning Algorithm

The concept behind tuning with generic algorithm is that you feed in proportional gain (Kp) and integral time (Ti) iteratively to the system and with each iteration you check the response. Finally, you will have the best possible gain values that is giving the best results you can have.

5.2.3.1. Inputs to the Generic Tuning Algorithm

In this generic algorithm, Input that are inserted to the model will be Kp and Ti. The range of kp with specific spacing and Ti as well. The input range is defined in MATLAB script that will be feed into Simulink model iteratively.

Range of Kp :

From 1 to 2 with spacing of 0.1

Kp input: 1, 1.1, 1.2 …….2

Range of Ti :

From 100 to 2000 with spacing of 30

Ti input: 100, 1.30, 1.60 …….2000

5.2.3.2. Output of the Generic Tuning Algorithm

The output will be generated in Simulink model with combination of Kp and Ti defined in the MATLAB input script and range is defined above. For each combination of inputs (Kp & Ti) the simulation will run and temperature is recorded for full simulation and that temperature values are feedback to tuning algorithm in MATLAB script that will calculate extent of deviation from the set point for each simulation and ultimately give us the optimum Kp and Ti values that give least deviation or smooth results near set point.

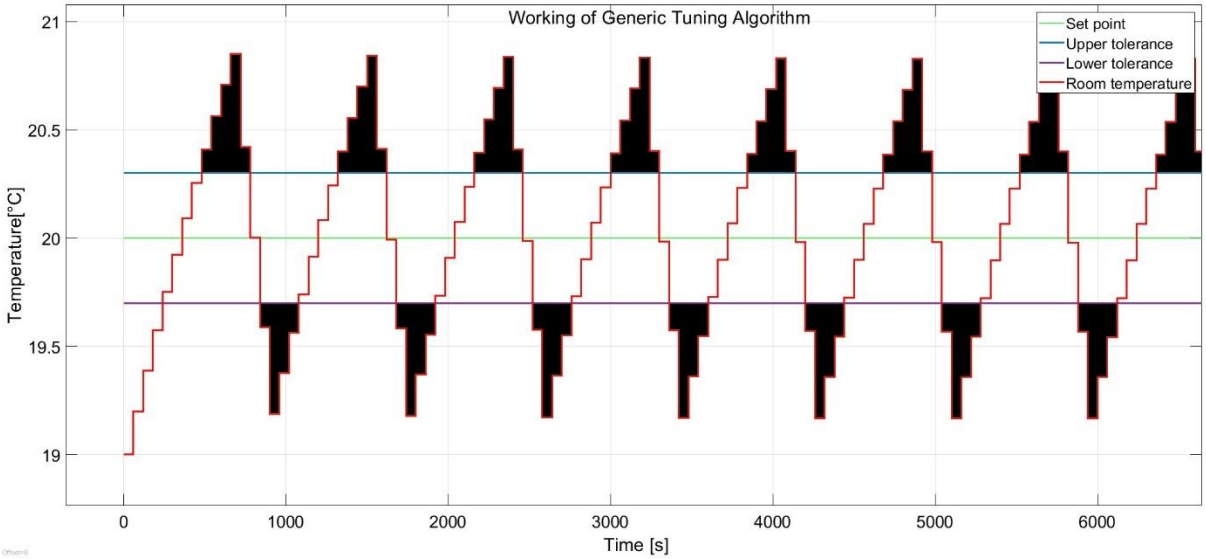


Figure 5.12: Working principle of generic tuning algorithm

5.2.3.3. Working Principle of Generic Tuning Algorithm

Let's consider that out response with some random Kp and Ti is in figure 5.12 .For example, set point is 20 °C , the tolerance value (temperature) is ± 0.3 °C around the set point. So, the upper tolerance value is 20.3 °C and lower tolerance is 19.7 °C. The concept is when the response is deviated more from these tolerance values for Kp and Ti values, consider it as a bad values of Kp and Ti.

Basically, the generic algorithm calculates the area that is covered as black which is the deviation from tolerance, it the area mentioned in black is high then those Kp and Ti gain values not suitable. On the contrary, the algorithm will give us those values which have minimum area under black. That means lower swings of temperature across set point.

5.2.3.4. Development Steps of Generic Tuning Algorithm.

- Find a way how to communicate Simulink and MATLAB and vice versa.
- From MATLAB to Simulink is easy just need to Create a workspace variable of Kp and Ti , that MATLAB variables can be read in Simulink.
- However, from Simulink, temperature response is to be recorded back to MATLAB script, for that there is Simulink block "back to workplace" , simple connect the Troom to it and set the values as show in figure 5.13

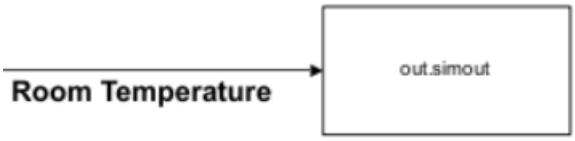


Figure 5.13: Simulink block (to the workspace) that send the signal values(temperature) back to MATLAB script either in array or time series form

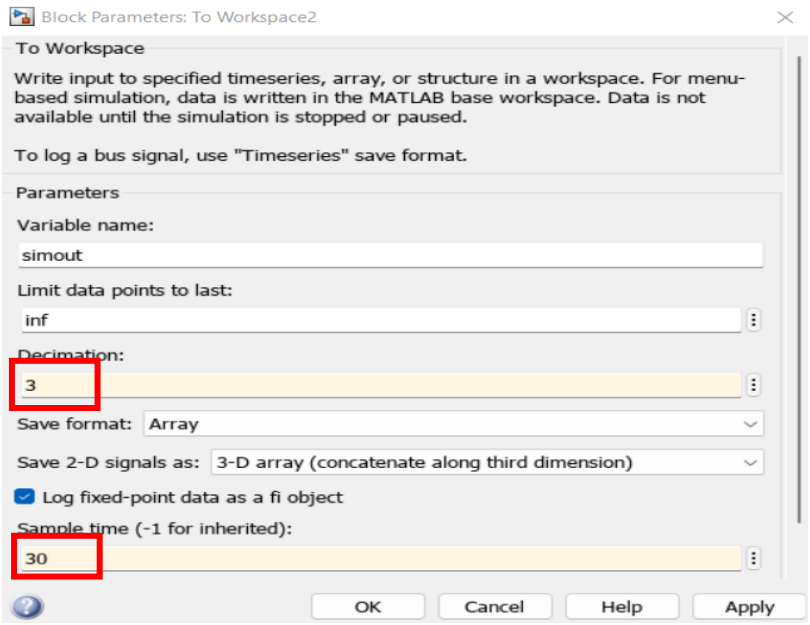


Figure 5.14: Interface of Simulink block (to the workspace)

- After all is set, just play the Run the script below, the script will feed to Kp and Ti values into Simulink model and get back temperature response.

```
clear
close all

%inputs
b=0;
kp_r = 2:.1:9.5                 % the range of values
%kp_r =6.3
Ti_r = 400:100:5000             % the range of values
%Ti_r = 3600

results = zeros(3,7)
for k= 1 : length(kp_r)
    kp = kp_r(k);
for n=1:length(Ti_r)
    Ti=Ti_r(n);

timet = [0:60:30000];
time = transpose(timet);
setPoint = 293;
star_temp = setPoint-2   ;
sim("Auto_tuning_model")         % this runs simulink model
temp = ans.simout;               % results from simulink model back to MATLAB script

% calcualtions for overshoot
overshoot_tol = setPoint+.3;    % 20+.3,,
overshoot_all = temp(temp > overshoot_tol & temp < overshoot_max); % all the elements more than 295.3
overshoot_all_minus = overshoot_all - overshoot_tol;
overshoot_sum = sum(overshoot_all_minus) % the response is best if this value is less
```

```
% calcualtions for undershoot
undershoot_tol = setPoint-.3;    % 20-.3,, -0.3 is the lower tolerance value
undershoot_min = min(temp);
undershoot_all = temp(temp > undershoot_min & temp < undershoot_tol); % all the elements
undershoot_all_minus = undershoot_all - undershoot_tol;
underhsoot_sum =-1*sum(undershoot_all_minus); % the response is best if this value is less
summ =overshoot_sum + underhsoot_sum;
b=b+1;


    results(b,1)=kp
    results(b,2)=Ti
    results(b,3)=overshoot_max
    results(b,4)=undershoot_min
    results(b,5)=overshoot_sum
    results(b,6)=underhsoot_sum
    results(b,7)=summ

end
end
```

Figure 5.15: MATLAB code of generic tuning algorithm

- Simulation will take some time, depend on you PC, after simulation is complete Run the code below, in command window that will give you tuned values of Kp and Ti.

```
>> clc
clear
load('slope_002710.mat')

last=results(:,7);
[value, index]= min(last(:))
full_row=results(index,:)
```

5.2.4. Case Study to Test Generic Tuning Algorithm

Let's create three cases in which our system parameters (Transmittance, external temperature, heating capacity) are different, but the rising slope is same. Then using the generic algorithm, optimum Kp and Ti will be calculated and use those tuned Kp and Ti values for other two cases and see either the response is similar or not. If the response is same that means whenever system has same slope or net heat gain response just insert those Kp & Ti values

Parameters	Case 1	Case 2	Case 3
Heating Equipment	Radiator	Radiator	Radiator
Set point	21 °C	21 °C	21 °C
Start temperature	15 °C	15 °C	15 °C
External temperature	**0 °C**	**0 °C**	**5 °C**
Transmittance (U)	**0.240**	**0.185**	**0.25**
Heating capacity	**65 °C**	**60 °C**	**60 °C**
Slope	**0.0040**	**0.0040**	**0.0040**
Proportional gain (Kp)	**0.5366**	**0.5366**	**0.5366**
Integral gain (Ti)	**808**	**808**	**808**
Temperature Differential	**±5 °C**	**±5 °C**	**±5 °C**

Table 5.3: Case study to validate the generic tuning algorithm

To sum it up, each slope will have respective Kp and Ti tuned gains. Now, we have three scenarios that gives us same slope or temperature gradient or net heat gain of the system, which is 0.0040.

Let's consider case 1 and insert the parameters in Simulink model Run the General Tuning algorithm which give us tuned values as

- Proportional gain (Kp) : 0.5366
- Integral gain (Ti) : 808

Now using these values proportional gain (Kp) and integral gain values (Ti), simulated the case 2 and case 3 and compare the results, if the results are more or less similar that means we can say that our approach of tuning is working perfectly.

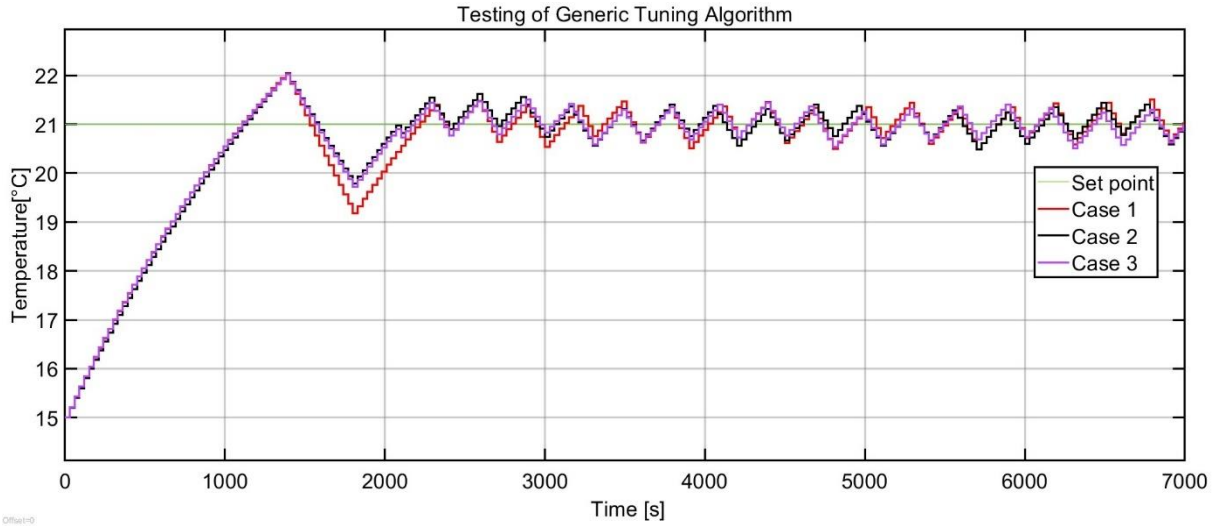


Figure 5.16: Temperature response validate the generic tuning algorithm

Comments on result

As expected, with tuned Kp and Ti values works for each case as the slope was same for each case. As you can see that temperature from 20 °C to 22 °C model calculates the slope, and each three cases are overlapping with each other. Moreover, when TPI control is in play, the temperature is overlapping using same Kp and Ti values although the systems parameters were different for each case.

5.2.5. Table of Kp and Ti

Now we need to make a table of slope and respective Kp and Ti. In order to do that, manually changing system parameters (Transmittance (U), external temperature (Texternal) , and heating capacity (Tmax) so that we have specific slope(eg 0.0030). After creating a system with specific slope, using generic tuning algorithm tuned the system in order to get tuned Kp and Ti for that slope.

Repeat the process, from slope range 0.0010-0.0070 and get the respective Kp and Ti. Here below is the complete tuned Table

Slope	Kp	Ti
0.0010	0.966	1100
0.0013	0.926	1070
0.0016	0.886	1040
0.0019	0.846	1010
0.0022	0.806	980
0.0025	0.766	950
0.0028	0.726	920
0.0031	0.686	890
0.0034	0.646	860
0.0037	0.606	830
0.0040	0.566	800
0.0043	0.526	770
0.0046	0.486	740
0.0049	0.446	710
0.0052	0.406	680
0.0055	0.366	650
0.0058	0.326	620
0.0061	0.286	590
0.0064	0.246	560
0.0067	0.206	530
0.0070	0.166	500

Figure 5.17: Tuned values of Kp & Ti

6 Development of Controller in Simulink

Until now we have tuned values of Kp and Ti, now we want out system to make it behave like we want to. As explained earlier, that we want to make system behave first ON/OFF and Then TPI. During ON/OFF cycle, the controller will tune itself by calculating the slope and finding the tuned Kp and Ti values from the Table 5.17.

After Kp and Ti are traced by the controller, it will insert the values onto TPI controller and ON/OFF relay is tuned OFF once TPI is turned ON.

In order to calculate slope/temperature gradient, controller need to detect only the first rising trend in which it can calculate the slope. Therefore, system should be designed in such a way it can detect 1st rising curve. However, the trend it follows depends on at what temperature the controller starts reading the temperature or when the system wakes up.

Therefore, in order to design the system which can distinguish all the scenarios. The controller is divided into three states namely State 1, State 2 and State 3.All these states are distinguished by The start temperature of the system.

***Start temperature = when system wakes up = 1st values recorded by sensor and given to controller**

6.1. States

States or scenarios define on base of first temperature recorded when system wakes up (thermostat turned ON) or when set point is changed. There were three scenarios hence divided into three states.

6.1.1. State 1

State 1 is called when start temperature (T_start) is less than the lower limit of the proportional band

Condition to enter State 1

If Tstart is < lower limit of PB

Controller behaviour in State 1

- Turn ON until it reaches upper limit of proportional band (Set point +PB/2)

  Troom ≤ (Set point +PB/2)
- Then Turn OFF until is it reaches lower limit of proportional band

Troom ≥ (Set point -PB/2)
(Slope calculation period)

- Exit from ON/OFF relay and enter TPI controller

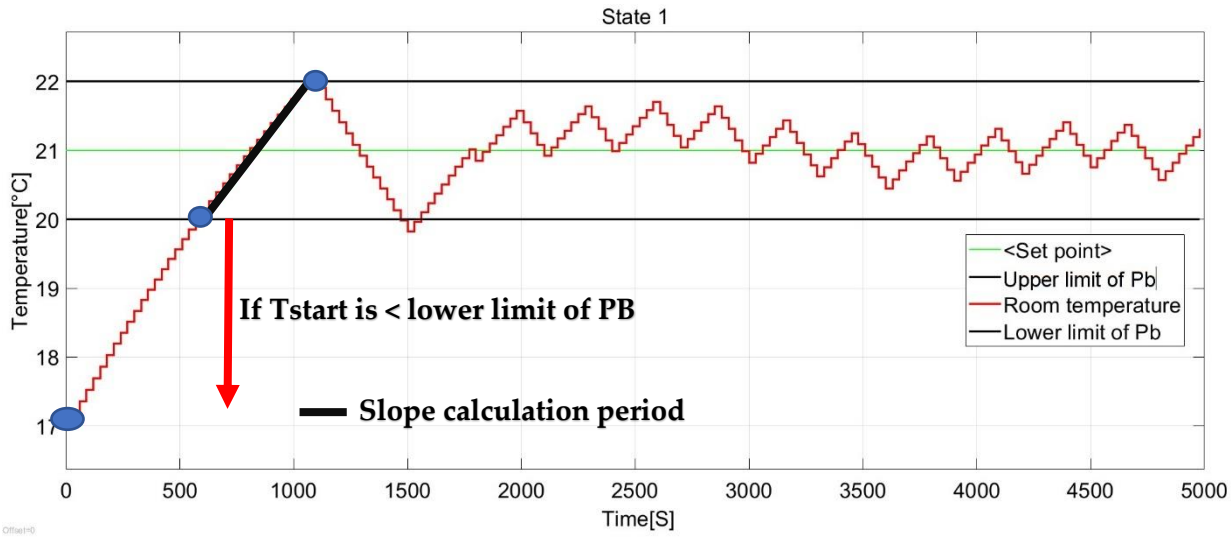


Figure 6.1: Demonstration of state 1 (If Tstart is < lower limit of PB)

As you can see in the figure 6.1 , that represent the State 1 Start temperature(T_start) is 15 °C and Lower limit of proportional band is 20 °C, so that's why it falls into state 1.

Furthermore, as upper limit of proportional band is 22 °C the system is ON until 22 °C and turned OFF until it reached 20 °C which is lower limit of proportional band. Once system hit 20 °C for second time, as switch in the model will switch the controller from ON/OFF relay to TPI .

6.1.2. State 2

State 2 is called when start temperature (T_start) is higher than the upper limit of the proportional band

Condition to enter State 2

If Tstart is > Upper limit of PB

Controller behaviour in State 2

- Turn OFF until is it reaches lower limit of proportional band
  Troom ≥ (Set point -PB/2)
- Turn ON until it reaches upper limit of proportional band (Set point +PB/2)
  Troom ≤ (Set point +PB/2)
  (Slope calculation period)

- Then Turn OFF until is it reaches lower limit of proportional band

  Troom ≥ (Set point -PB/2)
- Exit from ON/OFF relay and enter TPI controller

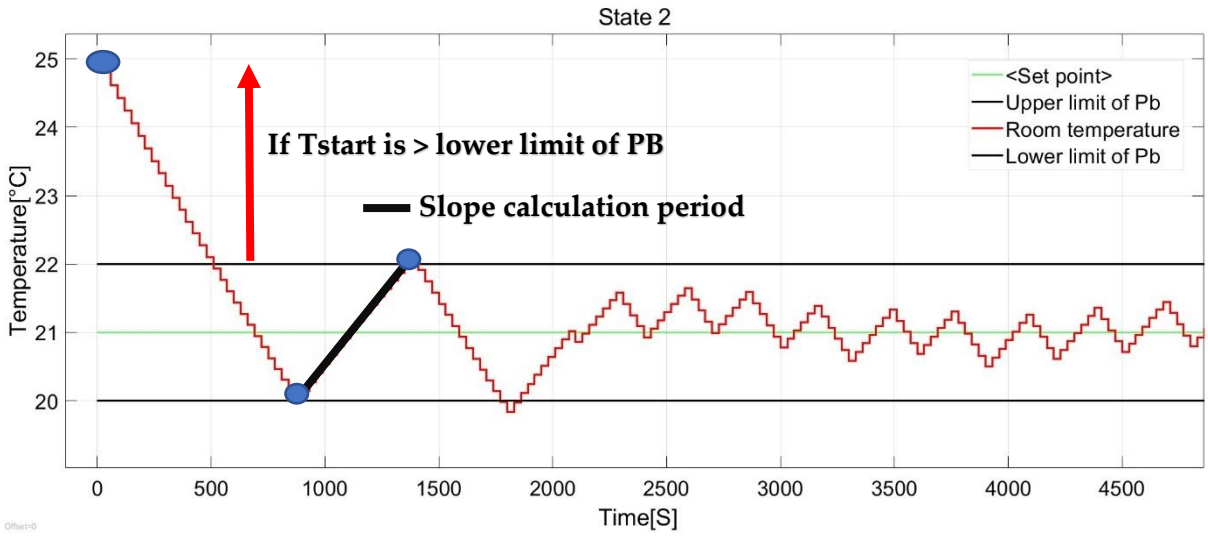


Figure 6.2: Demonstration of state 2 (If Tstart is > upper limit of PB)

As you can see in the figure 6.2 , that represent the State 2 Start temperature (T_start) is 25 and Lower limit of proportional band is 20 °C, so that's why it falls into state 2.

6.1.3. State 3

State 3 is called when start temperature (T_start) is higher than the upper limit of the proportional band

Condition to enter State 3

If Tstart is > lower limit of PB & Tstart is < upper limit of PB

Controller behaviour in State 3

- Then Turn OFF until is it reaches lower limit of proportional band

  Troom ≥ (Set point -PB/2)
- Turn ON until it reaches upper limit of proportional band (Set point +PB/2)

  Troom ≤ (Set point +PB/2)

  (Slope calculation period)
- Then Turn OFF until is it reaches lower limit of proportional band

  Troom ≥ (Set point -PB/2)

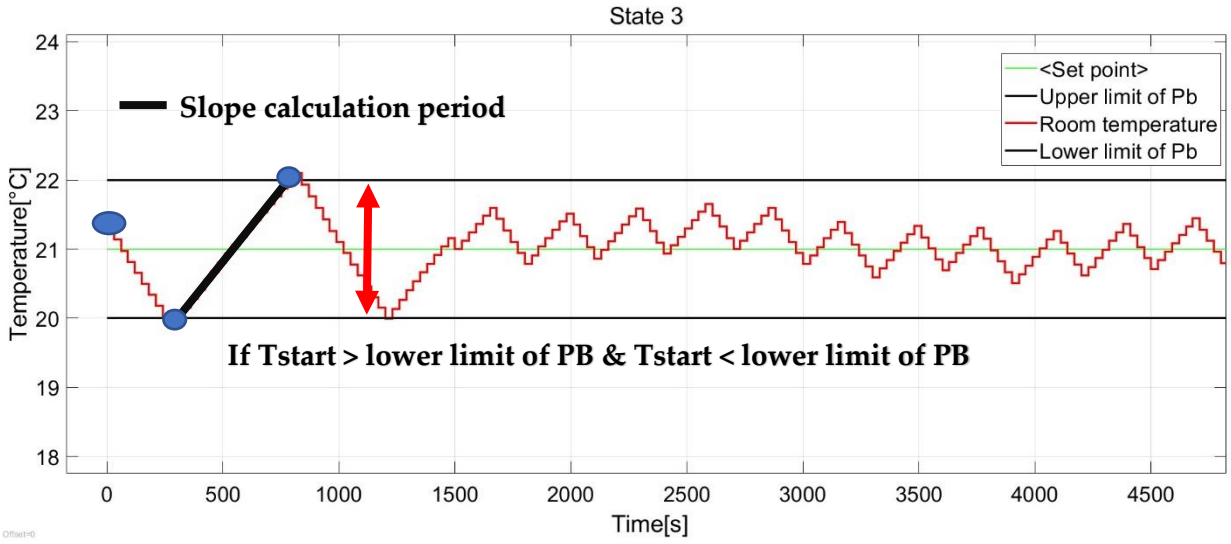


Figure 6.3: Demonstration of state 3 (within Pb)

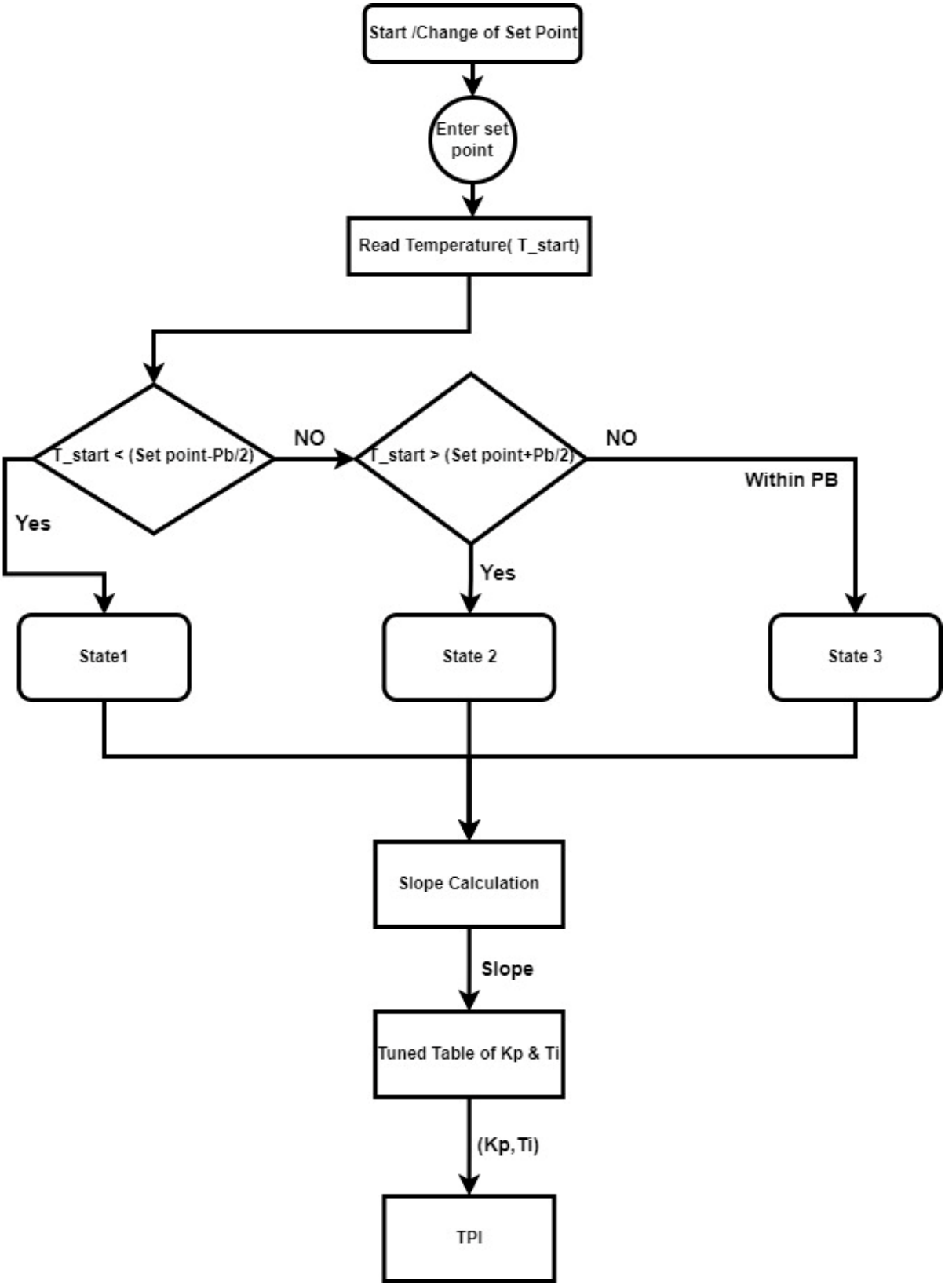


Figure 6.4: Flow chart for overall controller

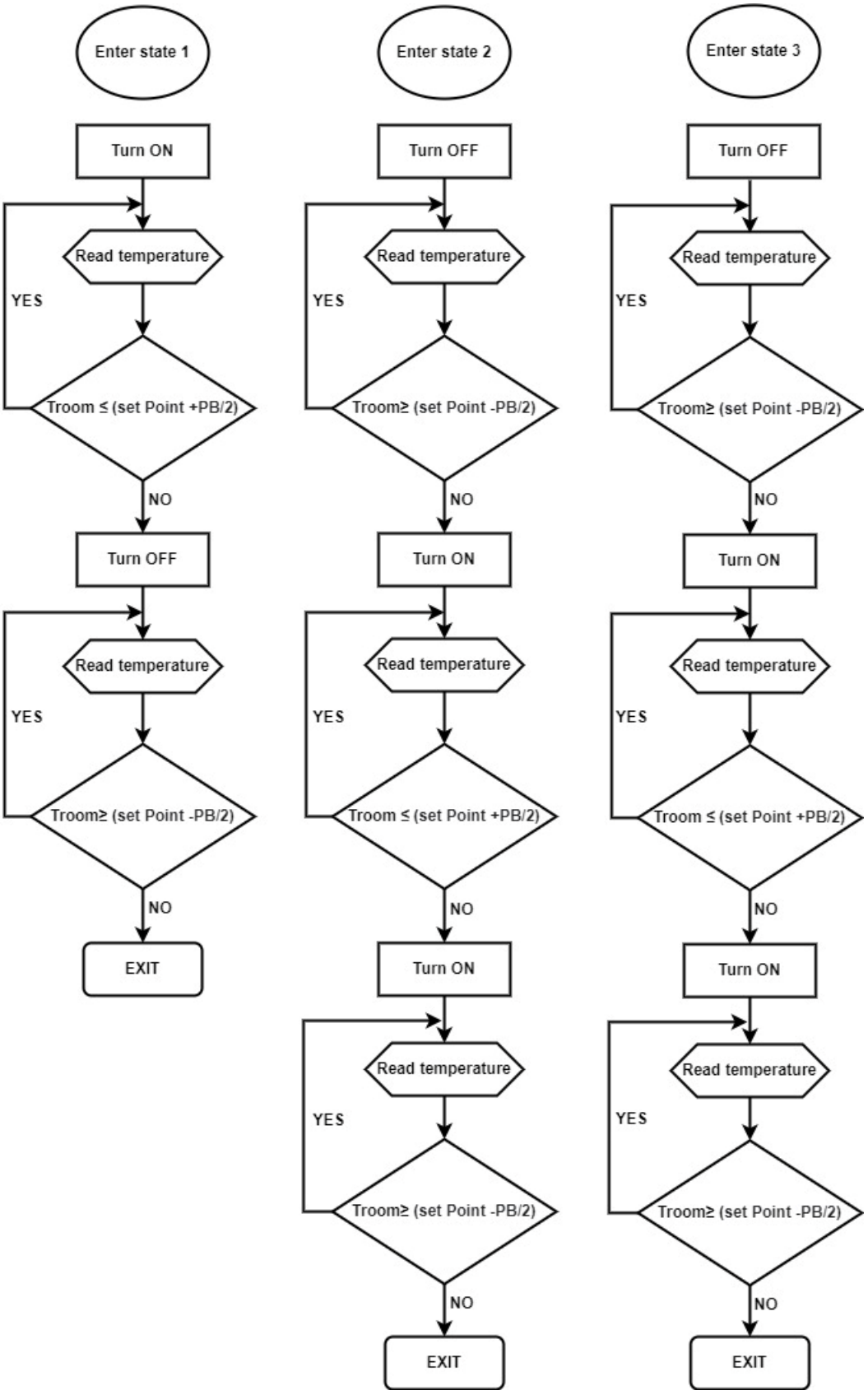


Figure 6.5: Flow chart of state 1, State 2, state 3

6.2. Final Simulink Model

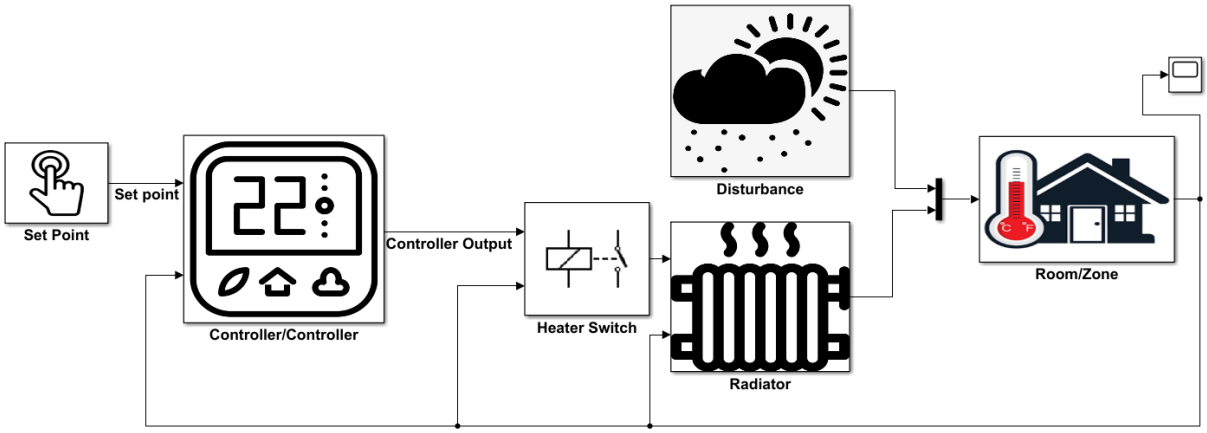


Figure 6.6: Final model of system

NOTE: The Model of room/zone, Heating device (Radiator and Fan coil), heater switch and disturbance are presented in chapter-3. However, in this chapter out main aim is to develop complete controller design.

6.3. Controller Design in Simulink

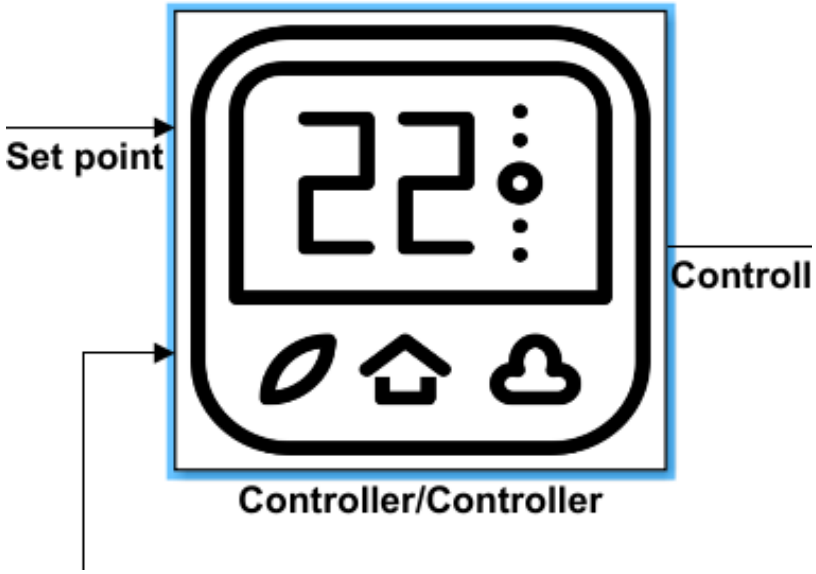


Figure 6.7: Simulink Controller subsystem

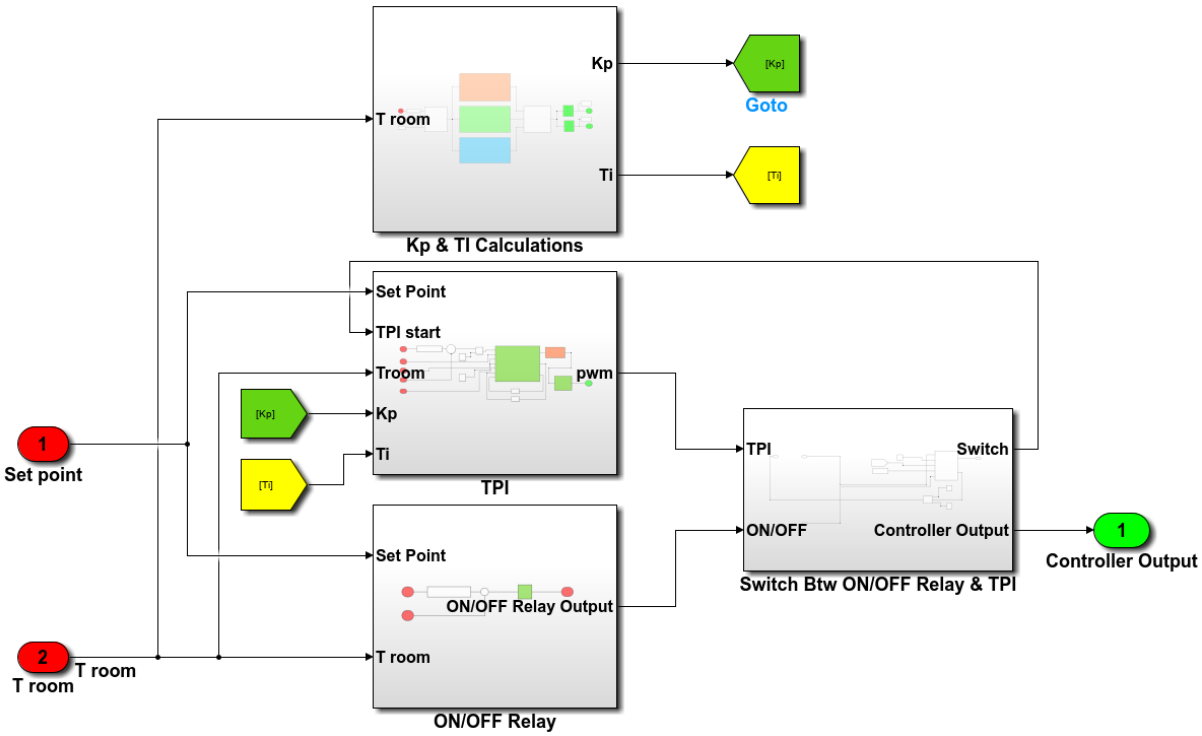


Figure 6.8: Inside of controller subsystem having TPI and ON/OFF relay controller connected in parallel

The figure 6.8 shows that there are four Subsystems inside the controller namely

- ON/OFF relay
- TPI
- Switch Between ON/OFF relay & TPI
- Tuning or Kp & Ti calculation

6.3.1. ON/OFF Relay

Among all the subsystem inside controller, it's the first one which will be active first, its basically ON & OFF operation half or proportional band above and half below the proportional band, the condition of ON and OFF is specified in Fig XX

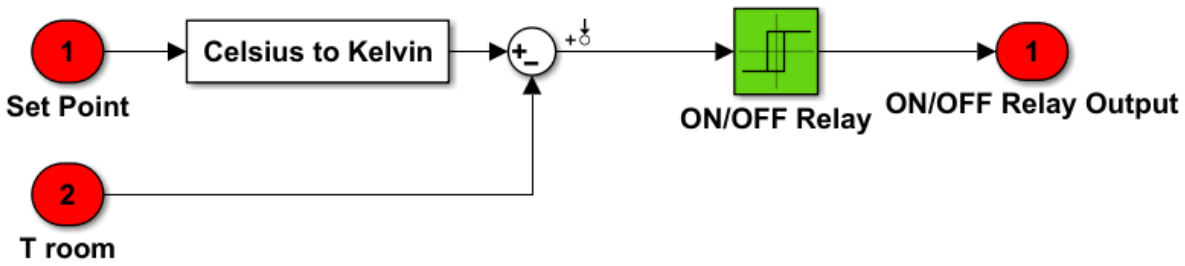


Figure 6.9: Simulink ON/OFF relay controller

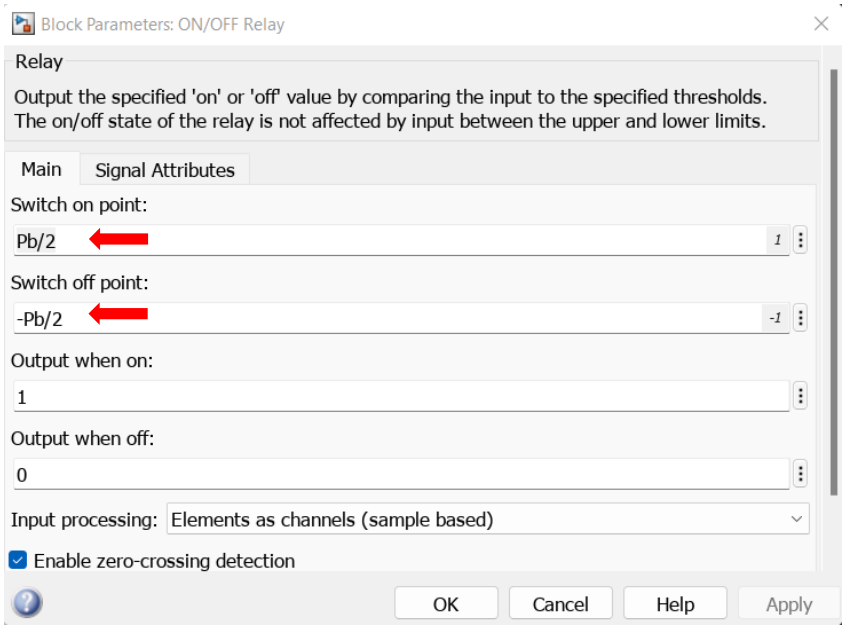


Figure 6.10: Simulink interface of relay block

6.3.2. Time proportional control (TPI)

Inputs to TPI

- Set point
- Kp and Ti from the tunning block
- Troom & TPI start (when to start TPI)
- Percentage error = (Toom – setpoint)/proportional band

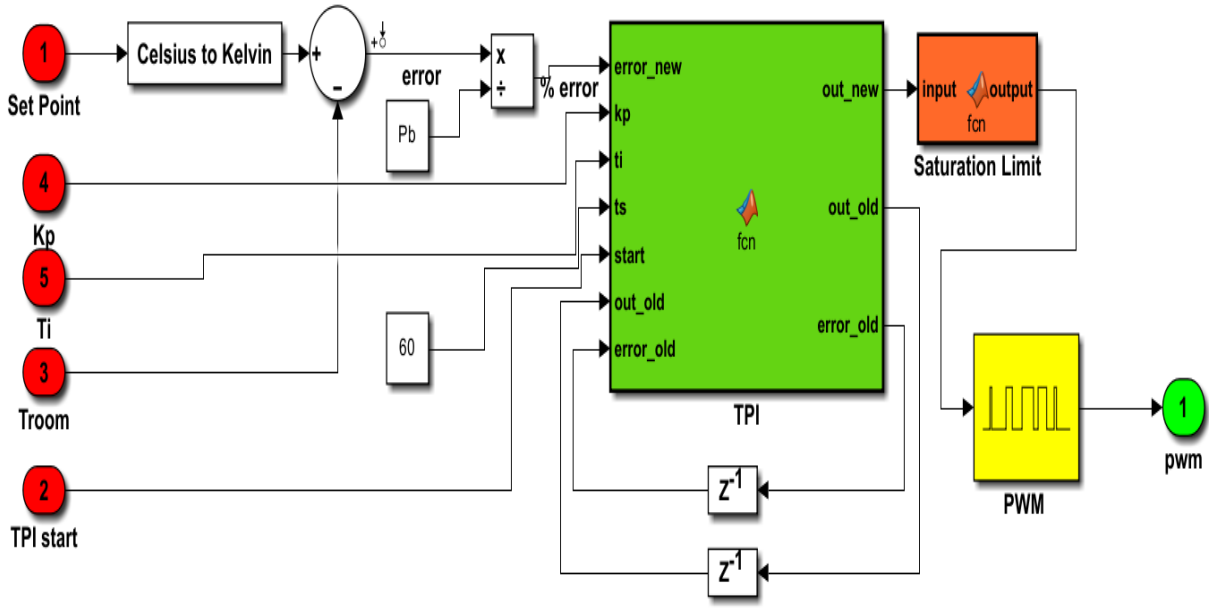


Figure 6.11: Simulink TPI subsystem

6.3.2.1. TPI Algorithm

It's the TPI algorithm that is also discussed in previous chapters as well , the MATLAB function in Simulink is show in the figXX below.

```
function [out_new,out_old, error_old]  = fcn(error_new,kp,ti,ts,start,out_old,error_old)


if(start == 1)
    out_new =0.5 + kp*error_new ;                                  % initialization
else
    out_new =out_old + kp*(error_new-error_old) + (kp*ts*error_new)/ti;      % main algorithm
end


error_old = error_new;
out_old = out_new;
```

Figure 6.12: TPI algorithm inside MATLAB function block in Simulink

6.3.2.2. Saturation

This block is used because some times , there is some noise from controller signal , control should not have output more then 100 % or lower than 0 %, hence this block is used with simple logic condition.

```
function output = fcn(input)

if(input>1)                                %Saturation upper limit
    output = 1;

elseif(input<0)                            %Saturation lower limit
    output = 0;
else
    output = input
end
```

Figure 6.13: MATLAB function block in Simulink for saturation

6.3.2.3. PWM

In PWM, need to mention the cycle period in out case is 300 second (5 minutes) that is recommended for radiator and fan coil.

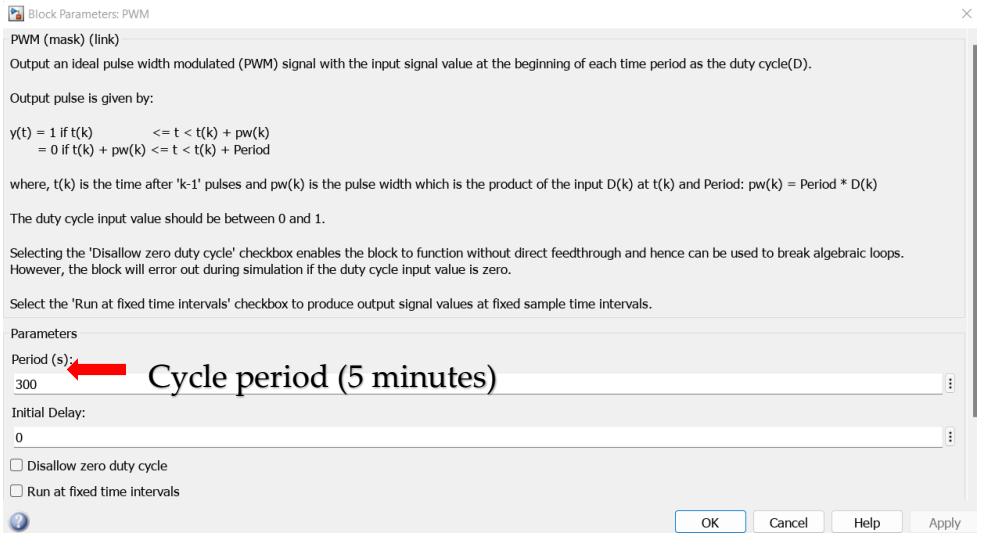


Figure 6.14: Interface of Simulink block for PWM mentioning 300 seconds cycle period

6.3.3. Switch Between ON/OFF Relay & TPI

It is switch design to switch between ON/OFF relay / TPI and the three states we defined before. This switch makes the system behave as we want it to behave, so that controller can do tuning and find gain values and feed in to TPI Controller.

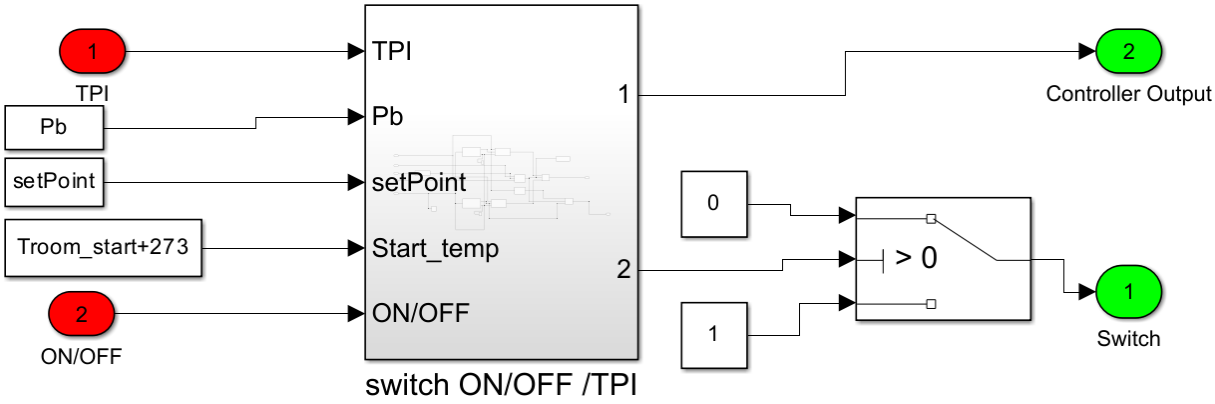


Figure 6.15: Simulink subsystem to switch between ON/OFF relay controller and TPI controller

Working principle of switch

The working principle of switch is to judge from the response/signal of ON/OFF relay which is either 0 or 1 as shown in fig XX. In Simulink model, Counter block is used which counts peaks are in the signal it could be either rising or falling. After the counter a simple switch block is used to trigger between ON/OFF and TPI.

State 1

The figure XX has response of ON/OFF relay and Temperature in one time frame. As can be observed that at first the output of ON/OFF relay 1 and then it become 0 and

then become 1, at this instant it gives us on rising peak, it's the point to switch to TPI. So, logic in switch is that if ON/OFF relay has 1st rising peak trigger the system to TPI.

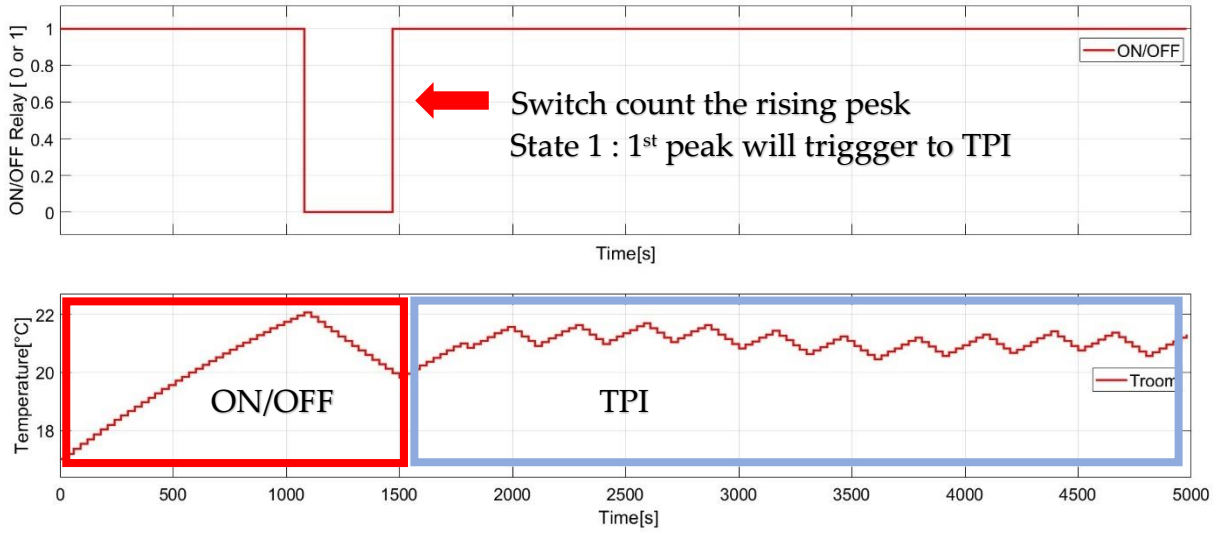


Figure 6.16: Principle of trigger system for state 1

State 2 and State 3

As far as state 2 and state 3 is concerned, the ON/OFF response has 2 rising peaks before switching. Therefore, in this case when counter has recorded 2nd peak trigger the system to TPI.

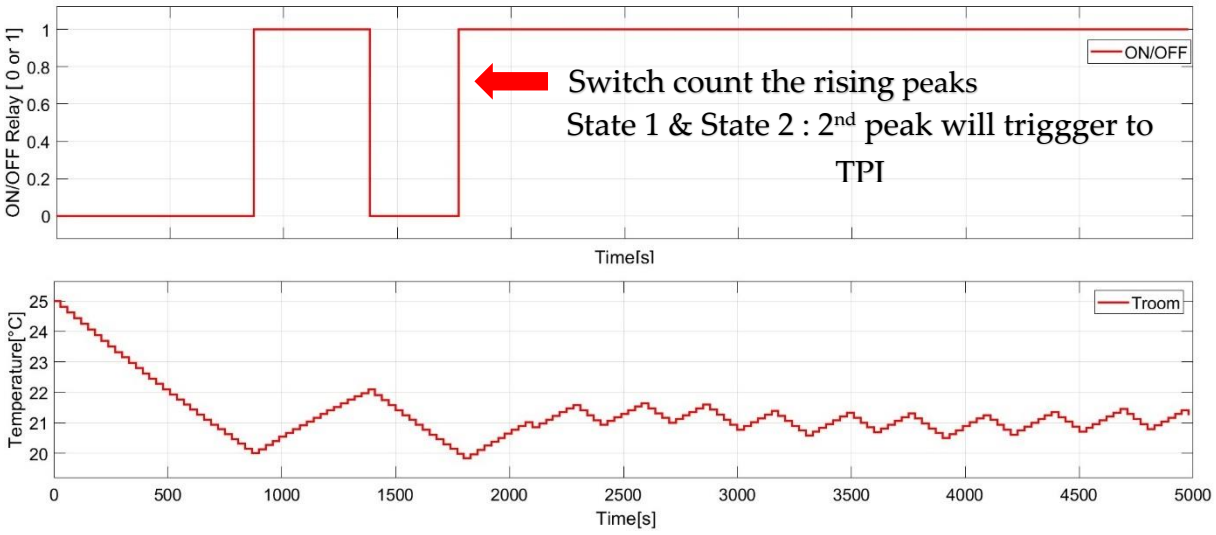


Figure 6.17: Principle of trigger system for state 2

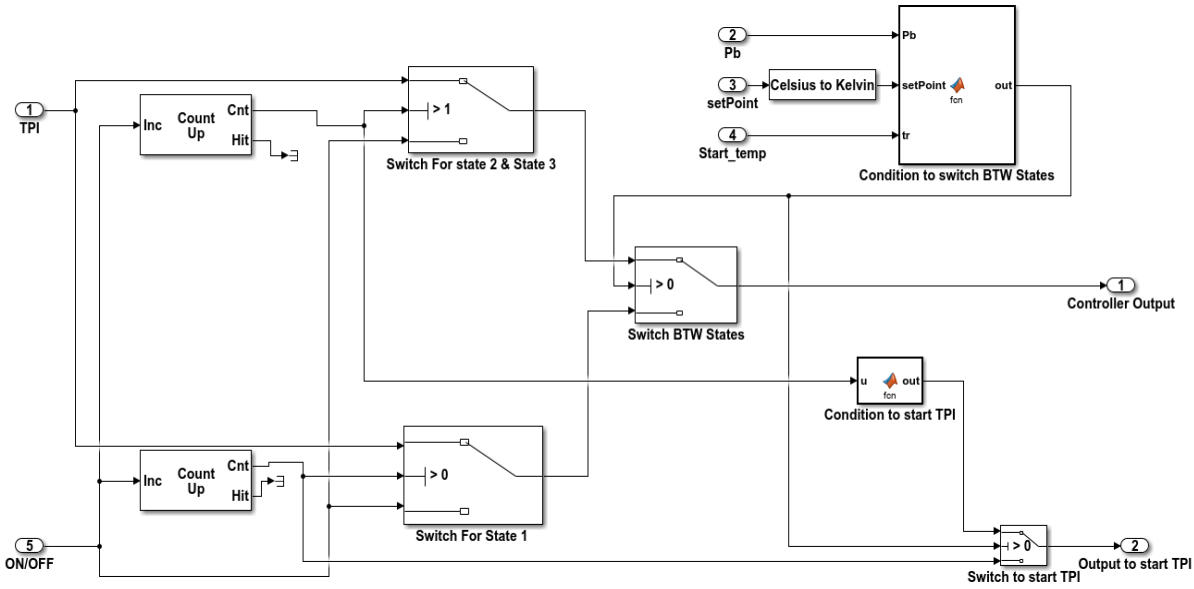


Figure 6.18: Inside of switch subsystem

Switch between States

Both state 1 and state 2 & state 3 are working is parallel, need to apply condition/ logic to distinguish between which one to use so a MALAB function block is shown in figure 6.19 is designed.

```
function out  = fcn(Pb, setPoint, tr)
out= 0
lower_Pb = setPoint-Pb/2;
if (tr> lower_Pb);
    out = 1;
elseif (tr < lower_Pb);
    out = 0;
else

end
```

Figure 6.19 : MATALB function block having logic/condition in it to switch between states.

Condition to start TPI

```
function out = fcn(u)


if (u ==2)
    out= 1
else
    out = 0

end
```

Figure 6.20: MATALB function block having logic/condition to start TPI

6.3.4. Auto Tuning / Kp & Ti Calculation

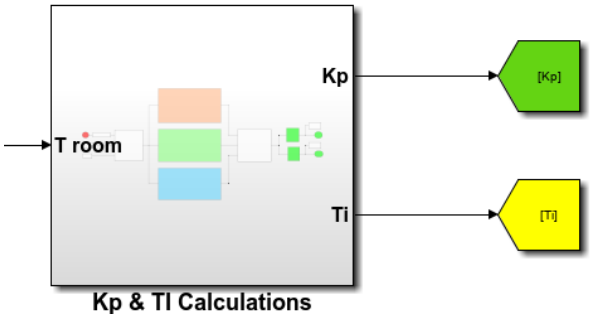


Figure 6.21: Auto-tuning subsystem

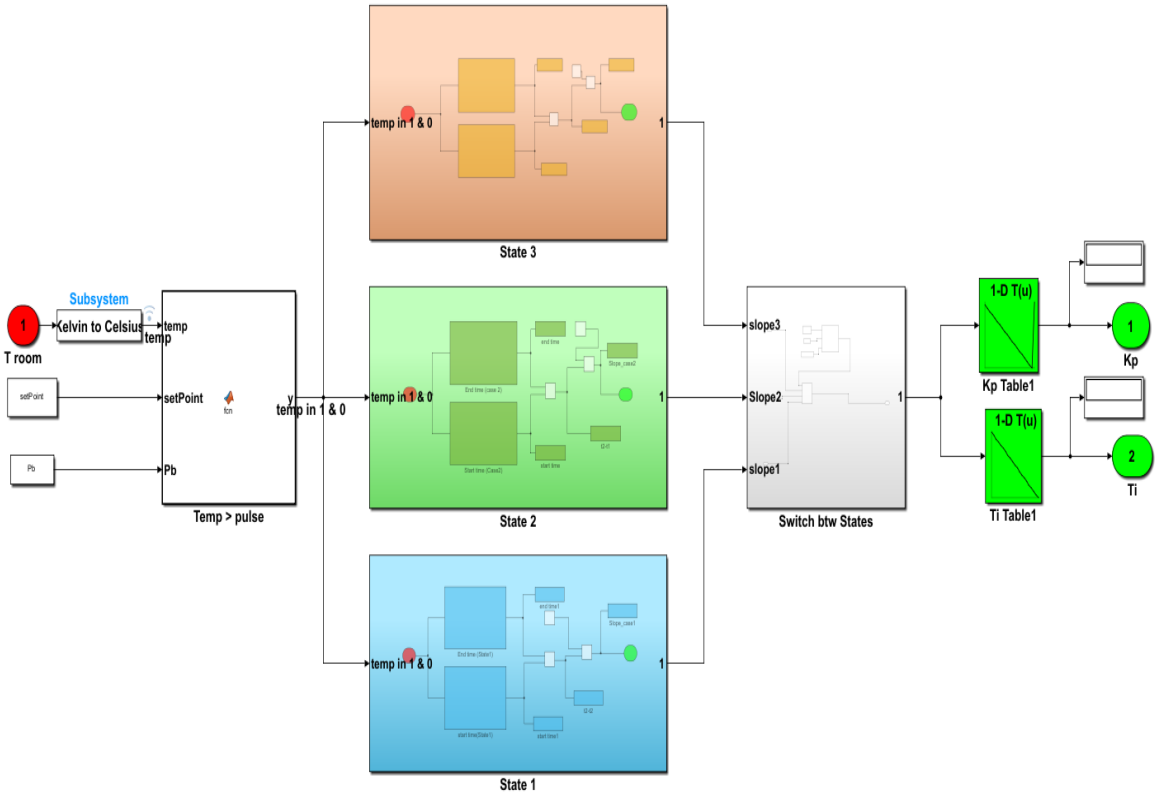


Figure 6.22: Inside auto-tuning block and having subsystem for state 1, state 2 & state3

6.3.4.1. Converting Temperature to 1 or 0 Pulses

The concept behind is that, with 0 and 1 pulse controller can record the rising or falling peaks of pulse , and on base of those peaks , time for slope calculation can be determined.

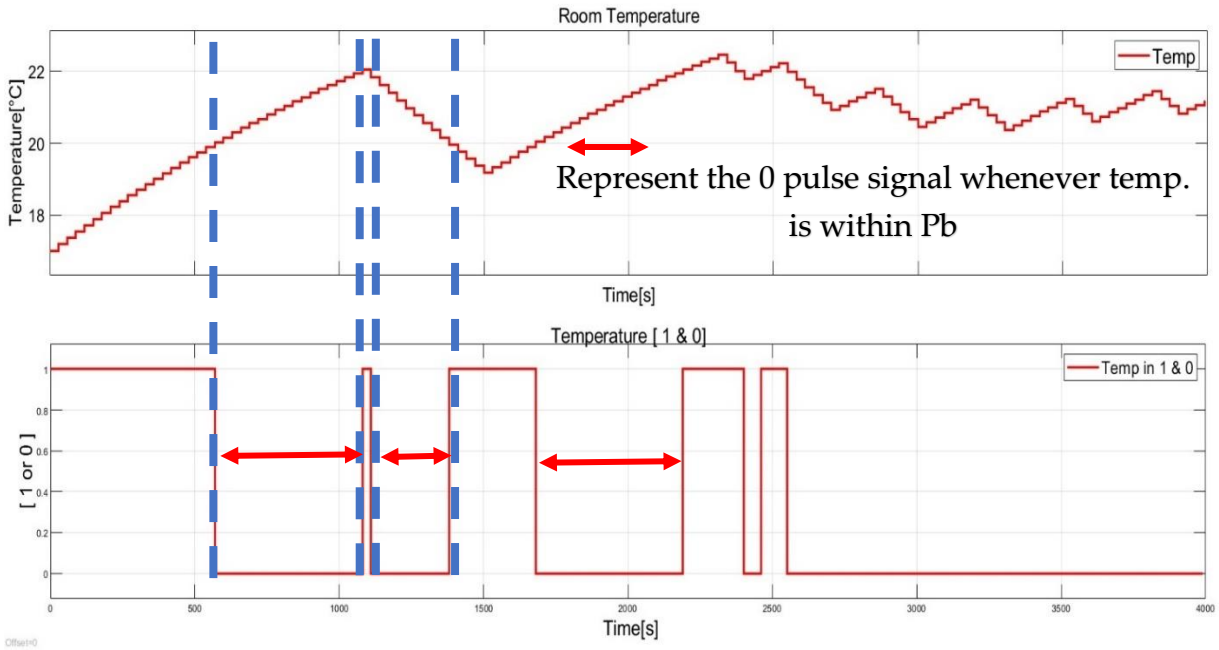


Figure 6.23: Concept of pulse generation from room temperature signal

```
function y = fcn(temp, setPoint, Pb)

Lower_Pb = setPoint-Pb/2;
Higher_Pb = setPoint+Pb/2;

if (temp <= Higher_Pb && temp >= Lower_Pb)
y= 0;
else
    y = 1;

end
```

Figure 6.24: MATLAB function/ condition to convert room temperature into pulse

6.3.4.2. Switch Between States During Slope Calculation

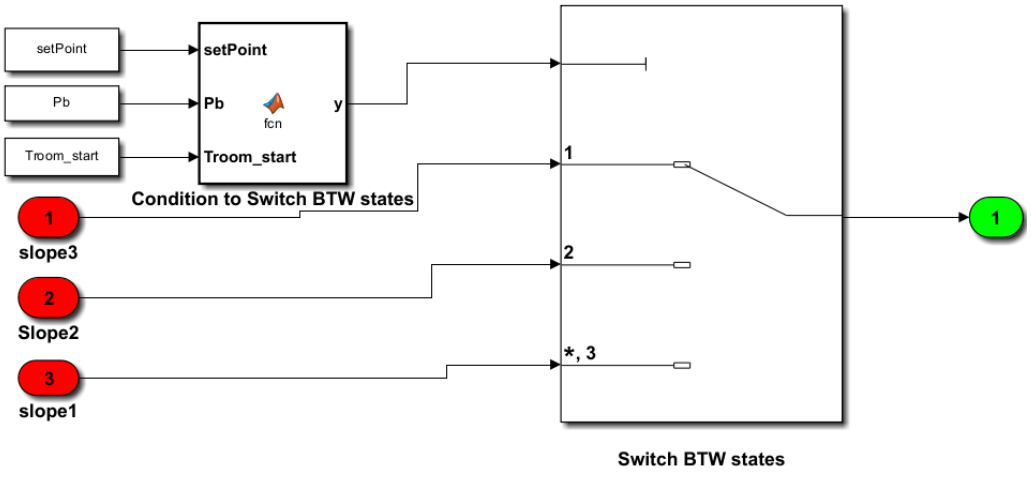


Figure 6.25: Simulink switch between states

```
function y = fcn( setPoint, Pb,Troom_start)

Lower_Pb = setPoint-Pb/2;
Higher_Pb = setPoint+Pb/2;

if (Troom_start < Higher_Pb && Troom_start > Lower_Pb)
y= 1;
elseif (Troom_start > Higher_Pb)
    y= 2;
else (Troom_start < Lower_Pb)
    y= 3;


end
```

Figure 6.26: MATALB Function/ Condition to switch between states

6.3.4.3. Slope vs Kp & Ti Table

In the lookup table tuned Kp and Ti values are inserted the input is the block is slope and it will give value of Kp and Ti

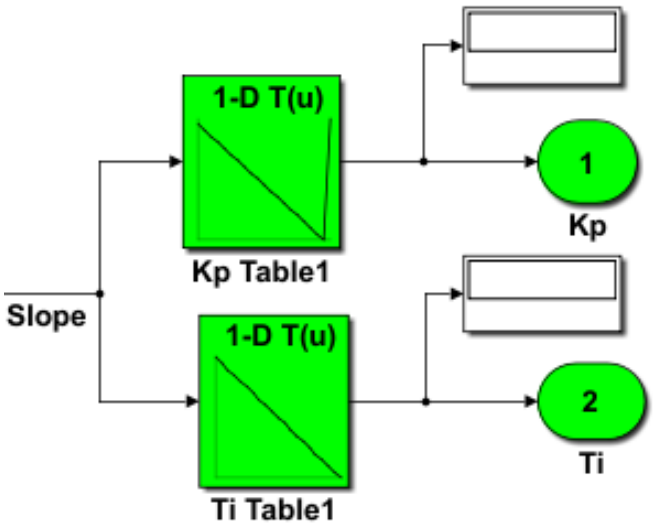


Figure 6.27: Simulink block (lock up table) for tuned Kp and Ti

6.3.4.4. State 1 slope Calculation in Simulink Model

For slope calculation, time is needed to calculate when temperature hit lower limit of proportional band (t1) and when it reaches upper limit of proportional band (t2). Here in the figure 5-44 can be seen that different subsystem is developed to dine the start time and end time and using below equation slope can be calculated.

$$\text{Rising slope} = \frac{Proportional\ band}{time\ from\ lower\ to\ upper\ limit\ of\ proportional\ band} = \frac{(T_2 - T_1)}{(t_2 - t_1)}$$

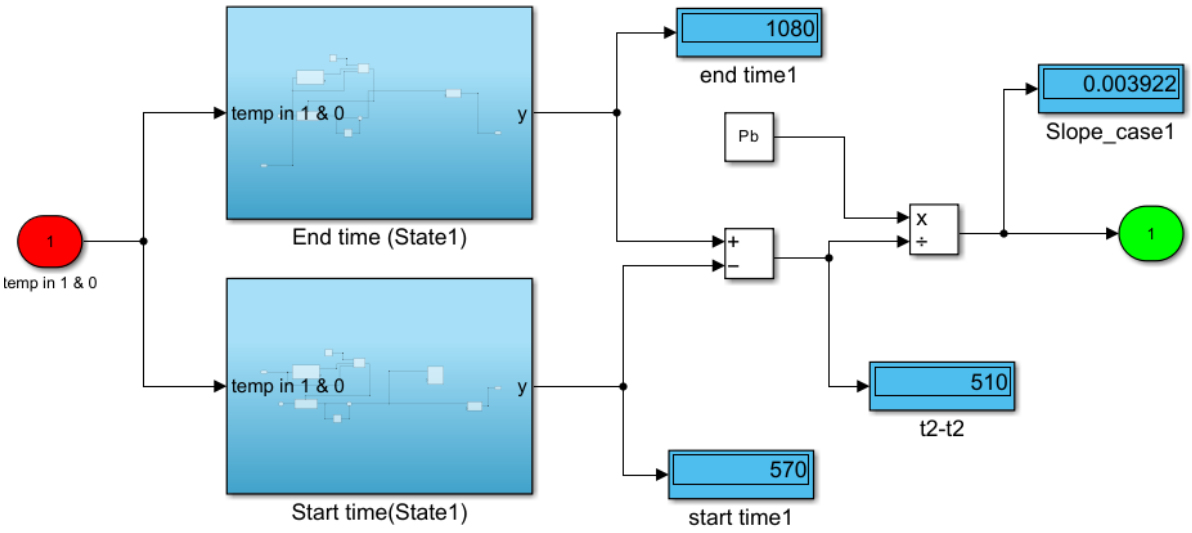


Figure 6.28: Slope calculation subsystem

Working of slope calculation

Temperature response is converted into pulse of 1 and 0. Start time and end time for slope calculation is determined by rising or falling peaks of the signal as show in the Figure 5-45, first falling peak is the start time and first rising peak(t1) is the end time(t2).In addition a trigger system is developed and a watch block is placed when system is triggered I gives the time when is got triggered and gives time either t1 or t2. Simulink subsystem is shown in detail in Figure 5-45.

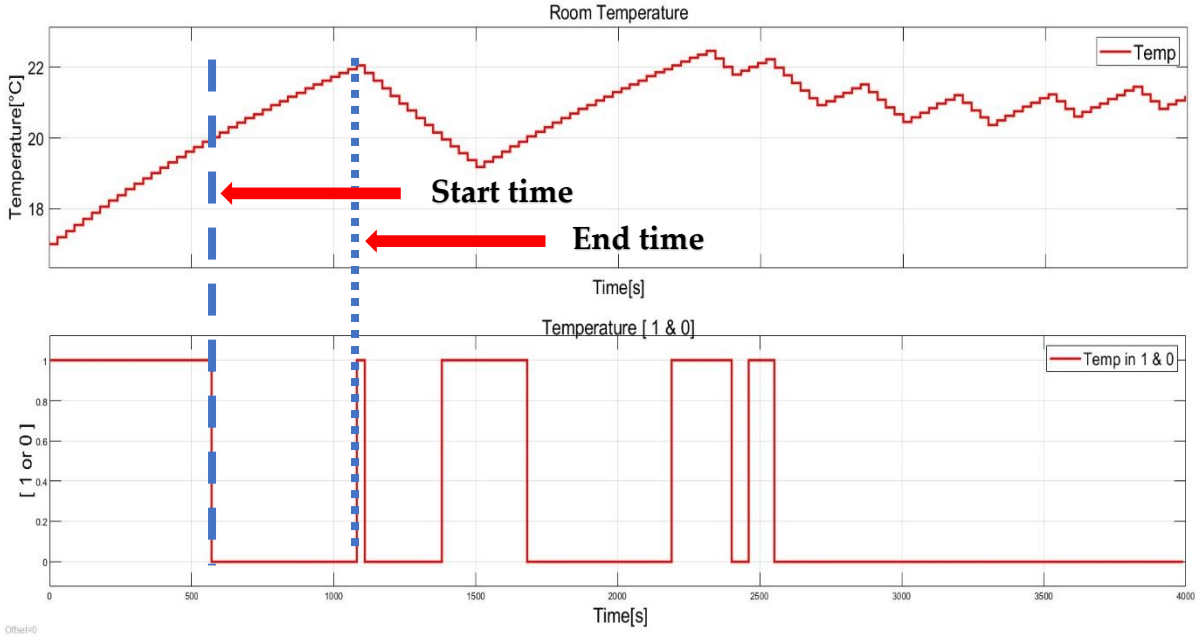


Figure 6.29: Concept of start time(t1) and end time (t2) for state 1, the falling and rising peak will me start time and end time respectively

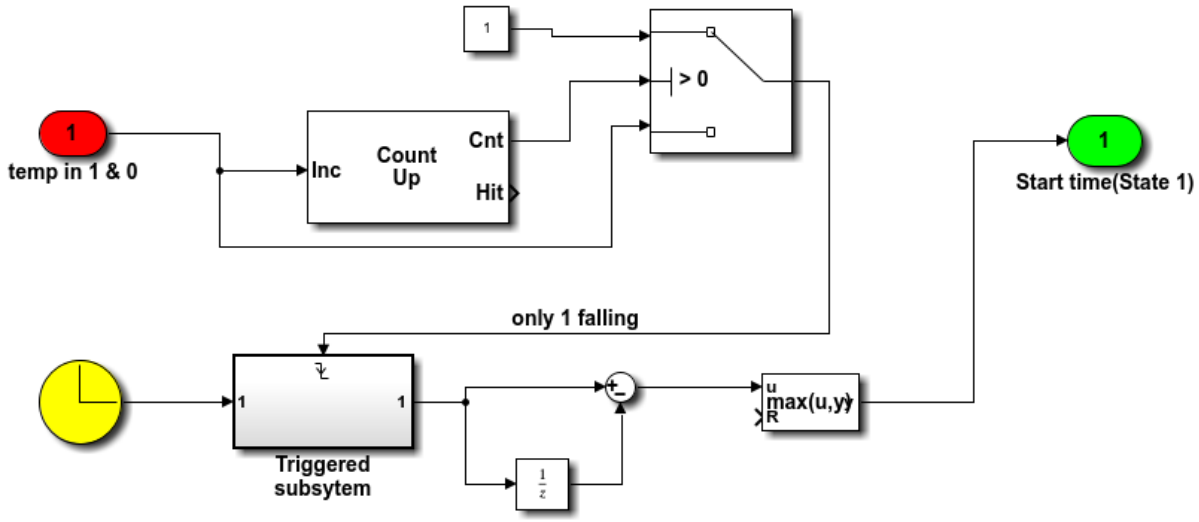


Figure 6.30: Subsystem to measure start time(t1) for state 1

Similarly, the end time is measured on base of same concept with off course with some modification in the Simulink model.

NOTE : Using the same concept ,for state 2 and state 3 different subsystems are developed in Simulink that have capability to calculate slope and hence Kp and Ti.

6.4. Advantages of Controller Design by this Approach

6.4.1. Fast Reach of Temperature to Set Point

This approach gives us best approach of temperature to set point in shorter period as possible. As explained before it first act as ON/OFF relay not TPI . If we use only TPI then it would have taken longer time to reach set point as in TPI controller, we set duty period in PWM where controller decide how much should be the next duty cycle (how much it should be ON in next cycle e.g. 60 %)

In case of only TPI and temperature is far away from set point TPI do not gives us 100 % duty cycle it could be 90 % or 80% and rest of the time it will be OFF. The TPI controller do that in order to avoid overshoot, but it will take longer time to reach set point. Hence this approach of using only TPI within proportional band gives us best possible response regarding the time taken by system to reach set point.

6.4.2. Minimum Overshoot

Moreover, by using this kind of approach for designing controller gives us minimum overshoot as possible. In this approach, TPI or PI only comes in play when temperature is within the proportional band. In PI controller, the integral term keeps on adding and as controller output is sum of proportional tern and integral term. Therefore,

temperature reach set point that sum of integral term show effect on output of controller and system experience overshoot.

In simple work later, controller turn on the TPI better it is. As in this approach, TPI is turn on after ON/OFF relay and system is near setpoint within proportional band.

6.4.3. Auto-tuning Capability

As explained in previous chapter, by using this approach we can auto-tune our system. ON/OFF relay tune the values and get the tuned Kp and Ti values and feed into TPI controller.

7 Simulation and Results

In the last chapter, model is developed and in this this chapter it's time to test or check the robustness of the model. In this chapter are reported and analysed the simulation results driven in different scenarios and changing the systems parameters or set point to validate the working of Simulink model that just developed.

7.1. Test 1 (Different State / Where System Wakes Up)

In this test, keeping the system same to check the result what if the system wakes up in three states as explained in previous chapters. Keeping the system same means, transmittance(U) / energy class, external temperature and heating capacity of heating device are kept same. The only difference will be the start temperature (Tstart) which defines out three states (state 1, state 2 , state 3), in this test we want to note that the temperature response is same, slope calculation by model is correct and as the system is to be kept same, the model has given the same Kp and Ti values or not.

Parameters	State 1	State 2	State 3
Heating Equipment	Radiator	Radiator	Radiator
Energy class	A2	A2	A2
Transmittance (U)	0.24	0.24	0.24
External disturbance / T external	0 °C	0 °C	0 °C
Heating capacity	**65 °C**	**65 °C**	**65 °C**
Proportional band (Pb)	**2 °C**	**2 °C**	**2 °C**
Set point	**20 °C**	**20 °C**	**20 °C**

Parameters	State 1	State 2	State 3
Start temperature	17 °C	12 °C	20.3 °C
Slope	0.0046	0.0044	0.0044
Proportional gain (Kp)	0.464	0.486	0.486
Integral gain (Ti)	724	745	745
Temperature Differential	**± 0.5** °C	**± 0.5** °C	**± 0.5** °C

Table 7.1: Data sheet for Test 1 (different states)

Simulation results

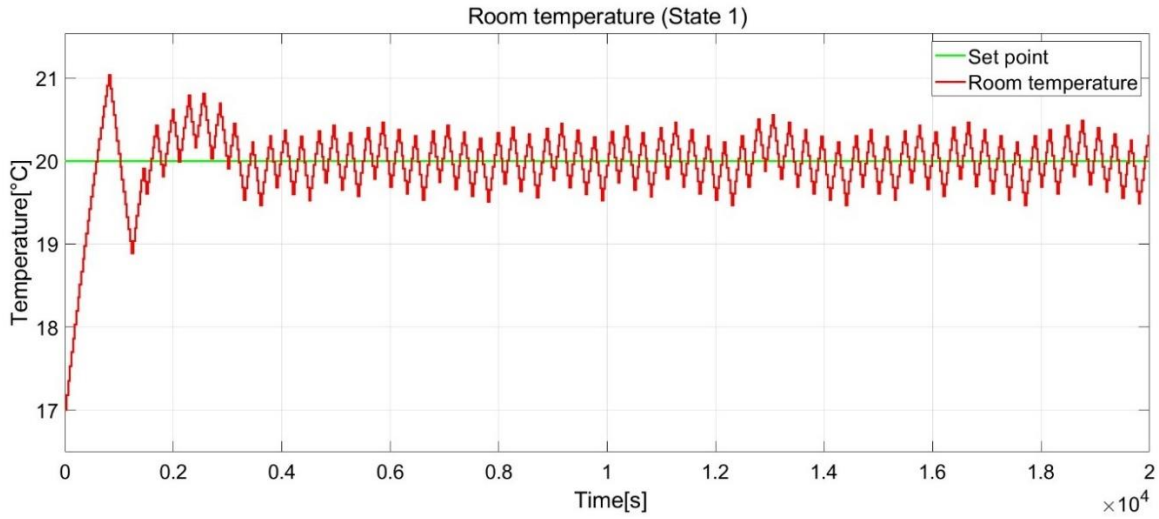


Figure 7.1: Simulation results for State 1

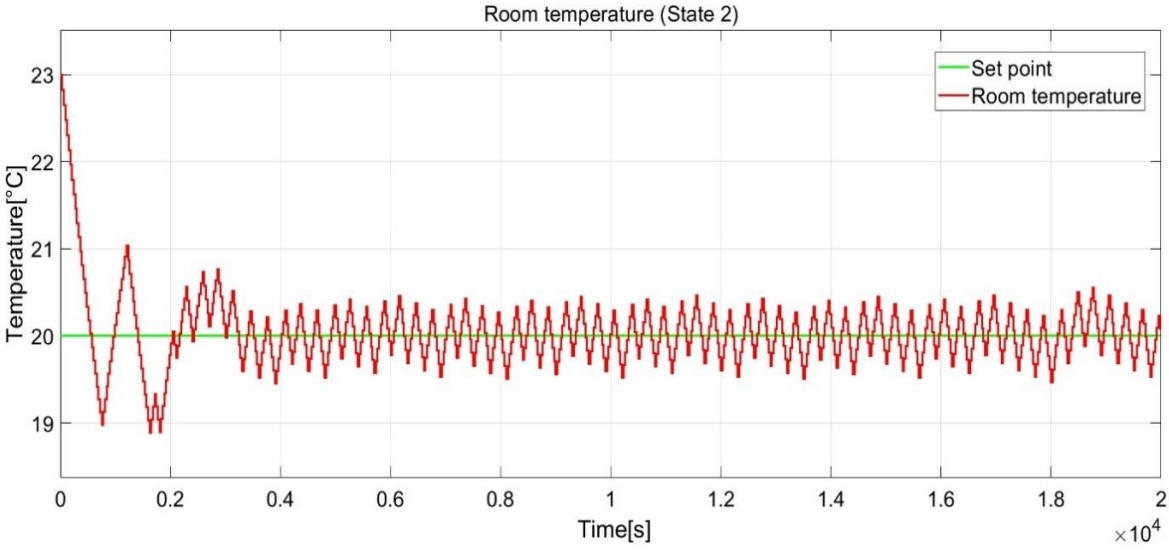


Figure 7.2: Simulation results for State 2

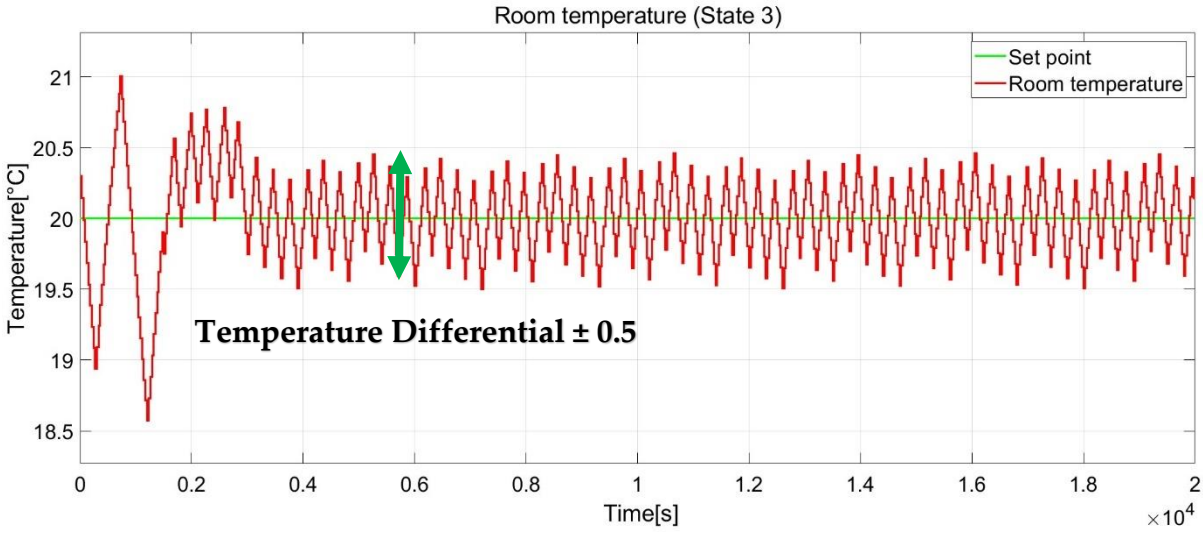


Figure 7.3: Simulation results for State 3

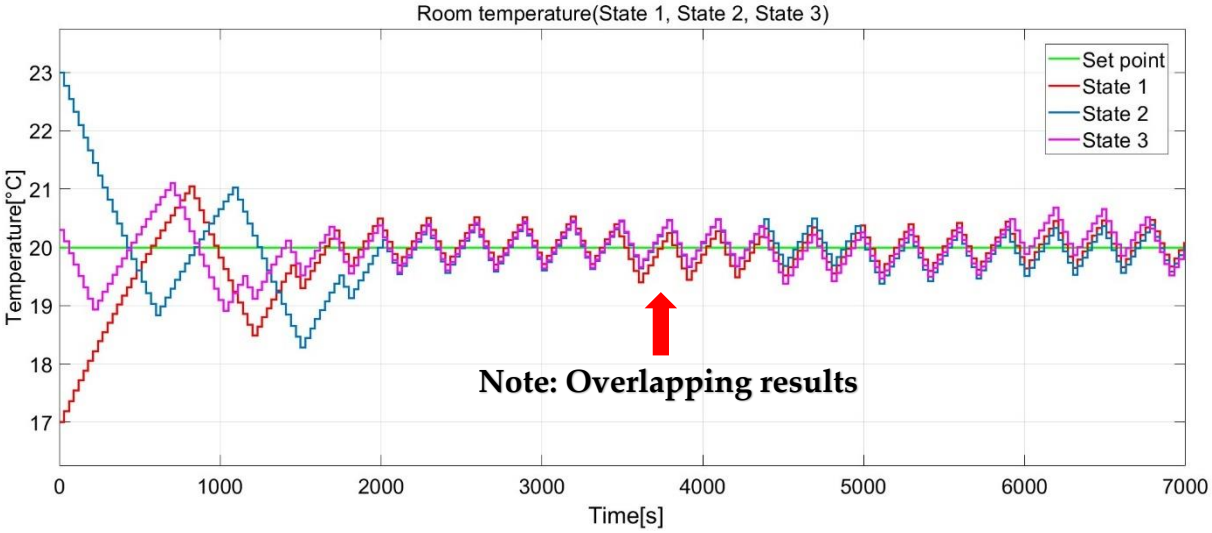


Figure 7.4: Simulation results for State 1, State 2, State 3 on one plane

Comments on result

As you can see that from figure 7.4 initially system was in different state therefore initially the temperature response is not overlapping once TPI comes in play the results are almost coinciding with each other. Therefore, that validates that the system wakes up in different state as the system parameters are same and temperature response is same hence model is working perfectly and can distinguish between states as well.

7.2. Test 2 (Different Set Point)

In this test, system parameters (Transmittance(U) / energy class, external temperature and heating capacity of heating device) is to be kept constant. Moreover, start time is kept same (same state e.g. state 1). Only set point will be changes in once case 20 and other case 22. It will be worth nothing that in both case although system parameters are same, but the Kp and Ti values should be same because around 20 and around 22 will be different from each other.

Parameters	Set point 20 °C	Set point 22 °C
Heating Equipment	Radiator	Radiator
Energy class	A2	A2
Transmittance (U)	0.24	0.24
External disturbance / T external	0 °C	0 °C
Heating capacity	65 °C	65 °C
Proportional band (Pb)	2 °C	2 °C
Set point	20	20

Parameters	Set point 20 °C	Set point 22 °C
Set point	20 °C	22 °C
Start temperature	17 °C	17 °C
Slope	0.0046	0.0037
Proportional gain (Kp)	**0.464**	**0.61**
Integral gain (Ti)	**724**	**830**
Temperature Differential	**± 0.5** °C	**± 0.5** °C

Table 7.2: Data sheet for Test 2 (different set point)

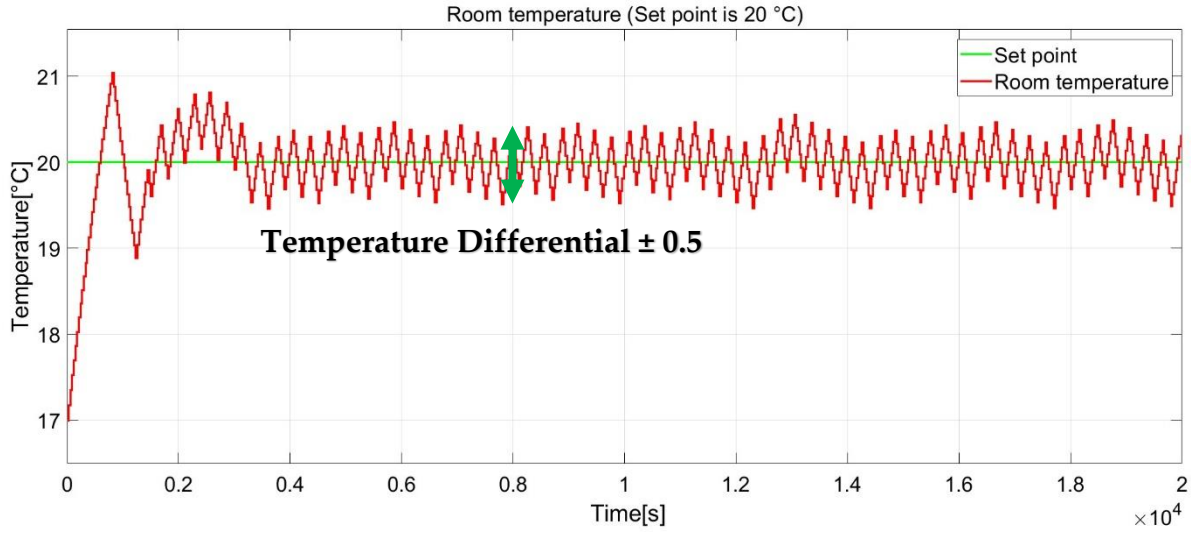


Figure 7.5: Temperature results at set point 20 °C

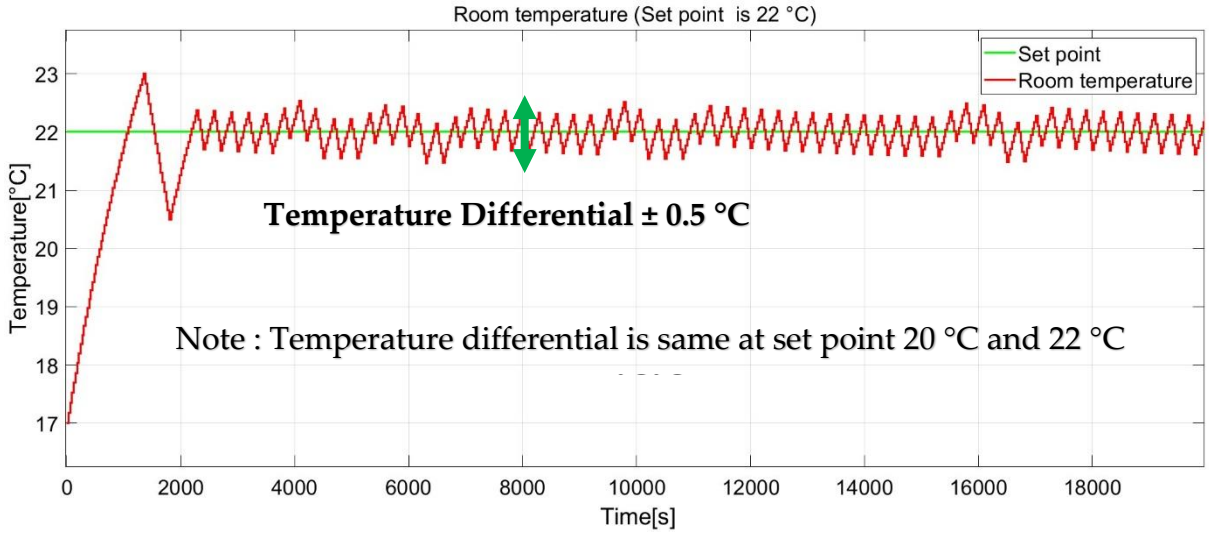


Figure 7.6: Temperature results at set point 22 °C

Comments

In this test, two things needed to observe one is that the even the system parameters are same, but the slope in each case should be different that can validated from table 7.2.The second one is that the temperature differential should be more or less same and form figure 7.5 and figure 7.6 shows that in both case the temperature differential is same ,±0.5 °C around set point.

7.3. Test 3 (Step Change in Set Point)

In this test, aim is to observe the temperature response after step change in set point, let's say, first set point is 20 °C and after some time set point is changed to 22°C.

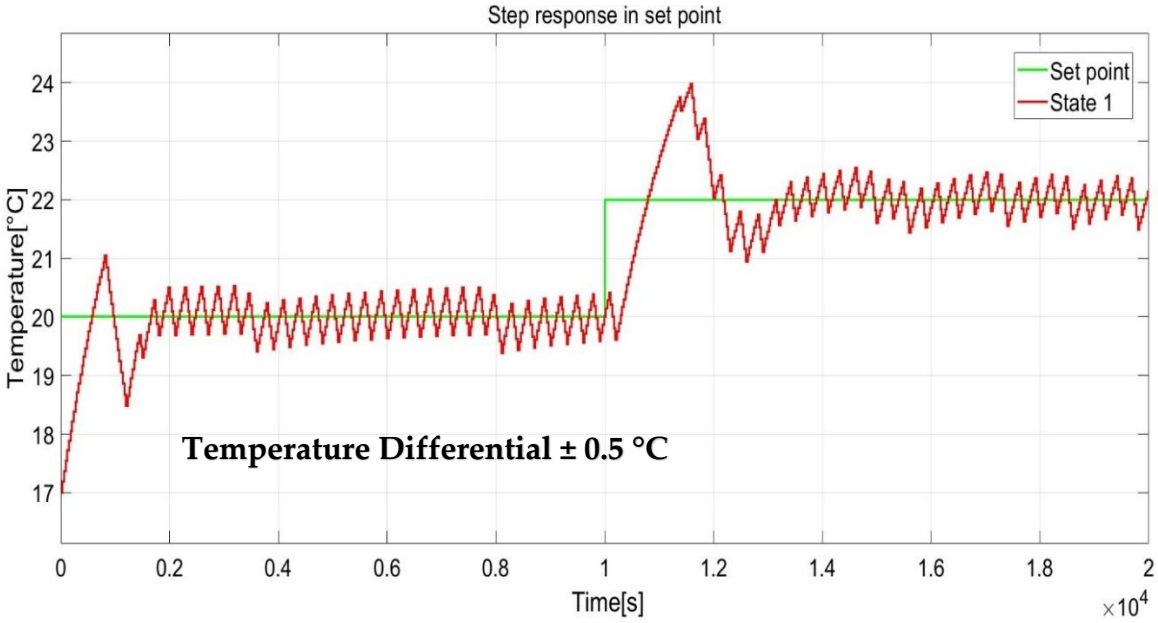


Figure 7.7: Temperature response by step change in set point

Comments on result

As can be seen from figure 7.7 the temperature differential is ±0.5 °C. hence the controller is working perfectly with exception of small overshoot initially and then temperature settle quite after some time.

7.4. Test 4 (Changing of Energy Class)

Let's try changing the system (transmittance or energy class) and keeping rest of the parameters same.

Parameters	A3	A2	A1
Heating Equipment	Radiator	Radiator	Radiator
Energy class	A3	A2	A1
Transmittance (U)	0.128	0.183	0.239
External disturbance / T external	0 °C	0 °C	0 °C
Heating capacity	**65 °C**	**65 °C**	**65 °C**
Proportional band (Pb)	**2 °C**	**2 °C**	**2 °C**
Set point	**20 °C**	**20 °C**	**20 °C**
Start temperature	**17 °C**	**17 °C**	**17 °C**
Slope	0.0033	0.0047	**0.0060**
Proportional gain (Kp)	**0.654**	**0.464**	**0.291**
Integral gain (Ti)	**866**	**723**	**594**
Temperature Differential	**± 0.5** °C	**± 0.5** °C	**± 0.5** °C

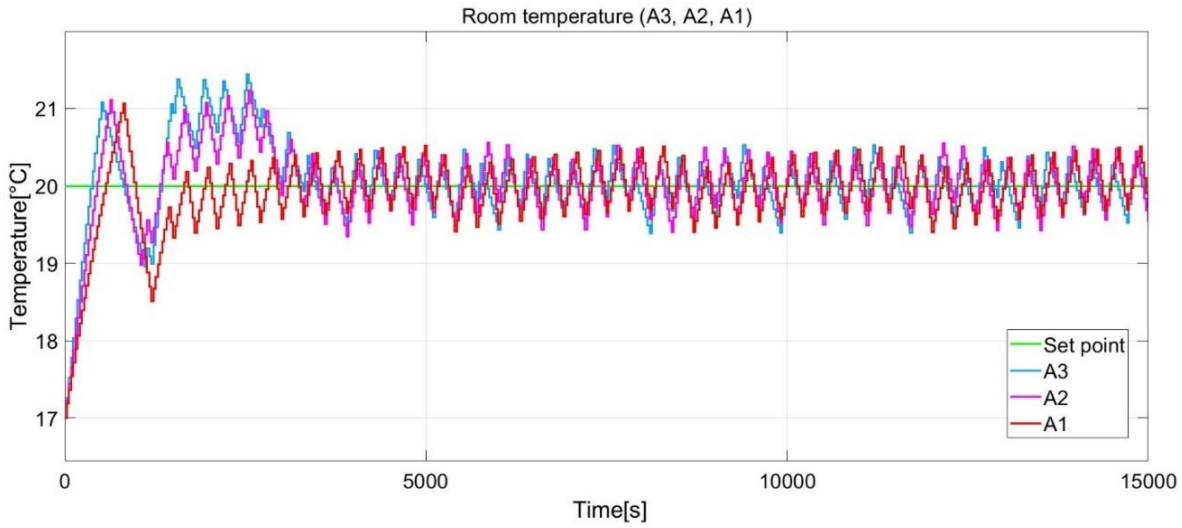


Figure 7.8 : Temperature response for different energy class (A3, A2, A1)

7.5. Test 5 (Random Changes in the System)

Parameters	Case 1	Case 2	Case 3
Heating Equipment	Radiator	Radiator	Radiator
Energy class	-	-	-
Transmittance (U)	0.15	0.20	0.40
External disturbance / T external	-10 °C	0 °C	+10 °C
Heating capacity	70 °C	75 °C	65 °C
Proportional band (Pb)	2 °C	2 °C	2 °C
Set point	20 °C	20 °C	20 °C
Start temperature	17 °C	17 °C	17 °C
Slope	0.0055	0.0051	0.0051
Proportional gain (Kp)	0.358	0.4156	0.41
Integral gain (Ti)	644	687	687
Temperature differential	± 0.5 °C	± 0.6 °C	± 0.5 °C

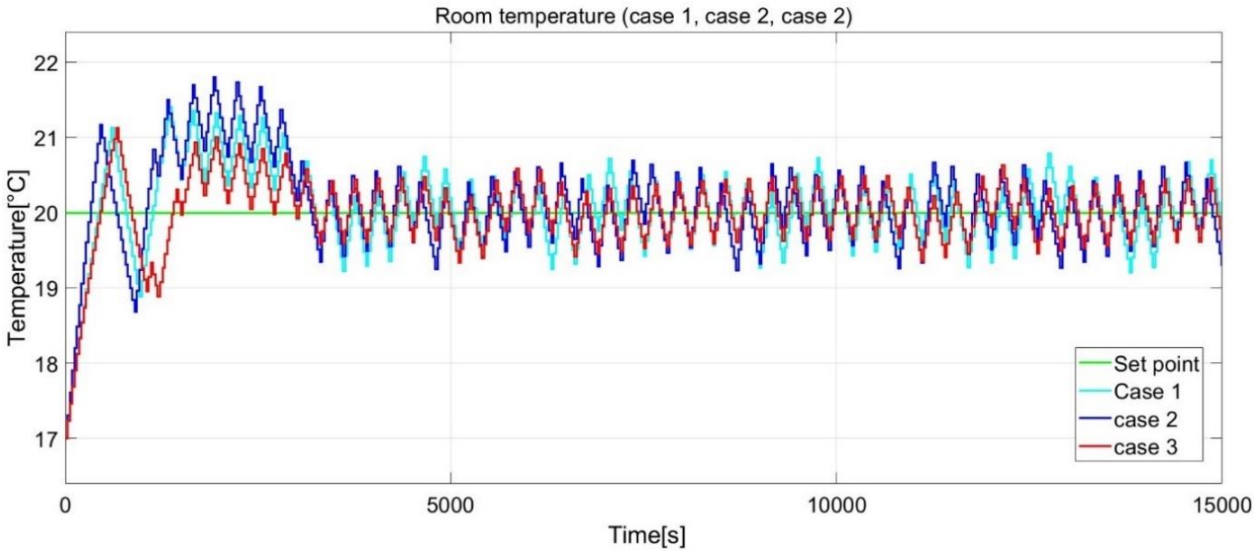


Figure 7.9: Temperature control for random change in system parameters

Comments on results of Test 4 & Test 5

As far as both tests are concerned, slope is different as the systems is different from each other, however with tuning the end response is same with same temperature differential/ temperature swings.

7.6. Test 5 (Continuous Disturbance / Change in External Temperature)

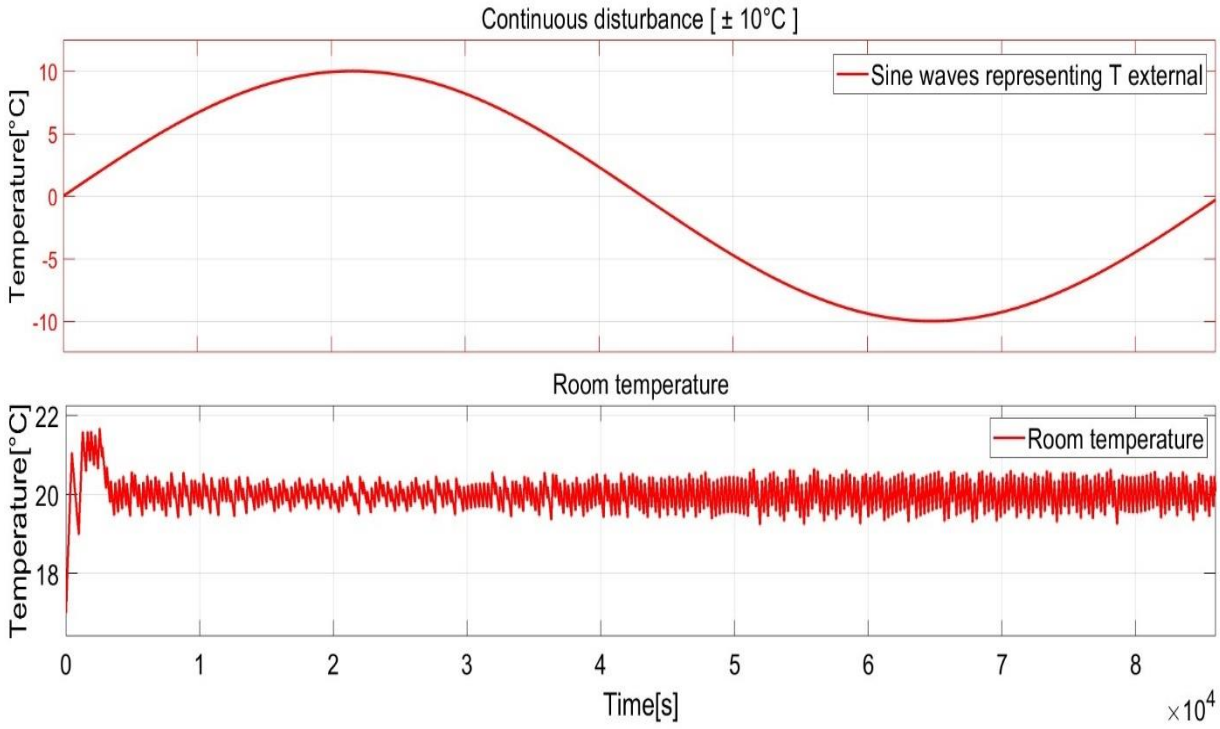


Figure 7.10: Effect on room temperature by continuous disturbance (External temperature)

Comments on result

The disturbance is created by continuous changing the outside temperature from -10 °C to +10 °C in 24 hours and simulation is runed for 24 hours and as you can see temperature response in figure 7.10 the temperature is pretty much constant around the set point. Therefore, this controller is robust enough to cope with external disturbance.

8 Conclusion

This project was about designing a temperature controller using the time proportional integral (TPI) method. In chapter 2, we set some objectives for the project and the most important target was to have a low-temperature differential (±0.6 °C). In order to achieve those goals, the model was developed in Simulink.

After the development of the model/simulation environment in Simulink, the temperature controller was developed and after the complete development of the controller, it was tested under different scenarios as explained in previous chapters. Some goals that were achieved are concluded here as follows

- The first aim was to have temperature swings of ±0.6 °C around the set point which is achieved and can be seen in figures 7.1 to 7.10 in each test control algorithm gave less than ±0.6 °C temperature differential.
- The controller performed well regarding reference tracking, as the temperature reached the set point quite fast thanks to the ON/OFF and TPI combination together. The controller is totally turned ON outside of the proportional band and only performs TPI within the proportional band that is the reason that desired temperature is achieved fast as possible.
- The controller is working and compatible with each energy class giving the desired temperature differential for each energy class as can be seen in figure 7.8.
- As the controller can tune itself to detect the heat gain and heat loss by the system hence it should be working perfectly regardless of the system. either the system/ room is well insulated, or the heating system is having more or less power. Whatever the parameter of the systems is the controller design is dynamic, and it works for all. This can be validated by results from figure 7.9 some random parameters were chosen but the controller behaved perfectly as the temperature response are overlapping.
- The controller is robust enough to cope with external disturbance as can be seen in figure 7.10.
- The overshoot is not that much in each test conducted. However ,only overshoot is observed in while step change in set point see the figure 7.7.

As far as future development is concerned, this model can be improved to make it even more precise like making temperature differential less than ±0.6 °C. Furthermore, another thing that could be improved is the cycle period, in our case we set it to 5 minutes (300 seconds) in PWM, we could try to make it a variable and try it to more than 5 minutes which would improve the life or relay and battery of the thermostat.

List of Figures

List if Tables

Bibliography

[1] Eurostate, "*Energy use in households in 2020*", Eurostate 2022.

[2] Christine Pout, "*BRE Client Report, Evidence Gathering - Compensation and TPI Heating Control*", September 2017

[3] GASTEC "*In-situ monitoring of efficiencies of condensing boilers – TPI control project extension*, GASTEC at CRE Ltd AECOM EA Technology September, 2010

[4] Autonics, "*Temperature control guide*" www.autonic.com

[5] P. Castro, A. Laso, M. Manana, A. Arroyo"*Smart Thermostats: An Experimental Facility to Test Their Capabilities and Savings Potential"* August 2017

[6] MathWorks " Control System Toolbox" The MathWorks, inc , 2022

[7] G. Hudson & C. P. Underwood " *A SIMPLE BUILDING MODELLING PROCEDURE FOR MATLAB/SIMULINK*" University of Northumbria

[8] L. Danza , L. Belussi, I. Meroni"*A simplified thermal model to control the energy fluxes and to improve the performance of buildings*" Italian Thermal Machines Engineering Association September 2016

[9] D T. Korsane, Y. Yadav, K. H. Raut*" PID Tuning Rules for First Order plus Time Delay System"* January, 2014

[10] ORMON "*Technical Explanation for Temperature Controllers*" www.omron.com

[11] P. Skruch "*A Thermal Model of the Building for the Design of Temperature Control Algorithms*" 2014

[12] Paolo Bolzern, Riccardo Scattolini" "*Fundamentals of Automatic Controls, Paolo Bolzern, Riccardo Scattolini*"

[13] M. G de Sousa neves"*RELAY METHOD ON AUTO-TUNING AUTOMATION SOLUTIONS*" Universidade Técnica de Lisboa, Instituto Superior Técnico, Lisboa, Portugal

Acknowledgments

I would like to thank Politecnico di Milano for initially giving me one of the best platforms and opportunities to study, explore and experience. Studying in one of the best institute in the world, a vibrant environment, and qualified professional certainly added value to not only my professional skills but also my interpersonal skills as well.

Moreover, I want to thank my family back in Pakistan who supported me in thick and thin, give me all the time to study abroad and complete my degree in the best way possible.

Lastly, my deepest gratitude to **Prof. Antonino Di Gerlando** for supervising my thesis from the Politecnico di Milano side and from the Fantini Cosmi side I would like to acknowledge the effort of Roberto **La Capruccia, Bianca Grazioli** & **Fabio Rodella** for guiding me throughout with their knowledge & wisdom for providing me all the time, direction and support needed to make this project possible.

This project is possible with the coloration of Politecnico di Milano and Fantini Cosmi S.p.a.

POLITECNICO
MILANO 1863