MODELLING AND SIMULATION OF A HEATING SYSTEM OF A SINGLE-FAMILY HOUSE

Faculty of Computer Science and Engineering Science
Cologne University of Applied Sciences

Submitted by:	Eduardo Guandulain Juarez	(11158980)
	Muhammad Munam Uddin Farooqui	(11160281)
	Nnamdi Ogbonnaya Odii	(11160333)
	Rohit Arunachalam Gopinath	(11158214)
	Yuganshu Wadhwa	(11158887)

Supervisor: Prof. Rainer Scheuring

Gummersbach, 17.07.2024

Technology
Arts Sciences
TH Köln

Contents

List of Tables

List of Figures

1 Abstract

This project focuses on the modeling and simulation of a heating system for a single-family house, utilizing an air-to-water heat pump as the energy source. The simulation is carried out using UniSim software. To enhance the model, system identification techniques were applied to the heat pump model provided in UniSim, facilitating the development of an equivalent model in Simulink. Pseudorandom binary sequence (PRBS) tests were conducted, and an advanced Proportional-Integral (PI) controller was developed. Communication between the PI controller and the UniSim model was established through Matrikon software, enabling the system to maintain the room temperature at a predetermined setpoint. The external disturbance, represented by the outside temperature (Tout), was modeled to reflect a 24-hour day-night cycle. Results demonstrate that, despite fluctuations in the outside temperature, the PI controller effectively manages the heating system to sustain the desired room temperature.

Similarly, a model predictive controller (MPC) was developed as a second approach using the same parameters. Its control performance was also tested using the plant in UniSim but only at a constant outside temperature. The room temperature was maintained for a short duration. Details of how both controllers were developed and tested are documented in this report.

2 Introduction

2.1 Heat pumps and their need

In order to mitigate the impacts of climate change, the global community is constantly exploring ways to cut down on greenhouse gas emissions. One effective strategy is the transition from fossil fuel dependency to the use of clean electricity. Within the residential sphere, a primary method of electrification involves the replacement of traditional oil, natural gas, or propane heating systems, as well as less efficient electric resistance heaters, with heat pumps. Heat pumps serve a dual function, acting as air conditioners during the hot months and, by reversing the refrigerant's flow in colder seasons, they extract heat from the exterior to warm the inside of a building. The process utilizes electricity for the mechanical transfer of heat rather than generating it directly, making heat pumps exceptionally efficient. This efficiency is reflected in the fact that the amount of heat delivered to space far exceeds the electrical energy consumed. Even when considering the lower efficiency of electricity produced by burning coal or natural gas compared to direct combustion in a home furnace, opting for a heat pump typically results in a net decrease in a building's greenhouse gas emissions [1].

2.2 Heat pump control

In recent years, there has been a notable surge in interest in employing technologies like model predictive control, big data analytics, and machine learning in the regulation of heat pumps. The integration of these technologies represents the state-of-the-art of HVAC (Heating, Ventilation, and Air Conditioning) technology, aiming at reducing energy consumption and CO_2 emissions while maintaining or improving comfort levels. The studies referenced below show various approaches, methodologies, outcomes, and comparative analyses of these technologies in application.

Authors	Title	Main findings
Mikkel Urban Kajgaard, Jesper Mogensen, Anders Wittendorff [2]	Model Predictive Control of domestic heat pumps. (American Control Conference, Washington, DC. 2013)	Evaluation of the potential economic benefits from adjusting the timing of the electrical heat pump load based on the fluctuating prices of electricity.
Thibault Q. Péan, Jaume Salom, Ramon Costa-Castelló [3]	Review of control strategies for improving the energy flexibility provided by heat pump systems in buildings. (Journal of Process Control, 2019)	Review of management control strategies for utilizing energy flexibility through heat pumps, distinguishing between rule-based controls and Model Predictive Control. The advantage of rule-based (RBC) strategies lies in their straightforward deployment, achieving respectable, though not perfect, outcomes. On the other hand,

Authors	Title	Main findings
		Model Predictive Control (MPC) requires more complexity and expense to incorporate, yet it delivers significantly improved performance.
Ali Etem Gürel, Ilhan Ceylan [4]	Thermodynamic analysis of PID temperature-controlled heat pump system (Karabuk University, Karabuk, Turkey, 2013)	The constructed heat pump dryer, regulated by a PID controller, underwent evaluation through the drying of mint, parsley, and basil. This evaluation encompassed assessments of energy, the ratio of energy utilization, exergy, and uncertainty. Within this system, the velocity of the drying air was adjusted based on the selected temperature. Consequently, as the temperature setting was raised, the speed of the drying air was correspondingly reduced.
Akmal, Muhammad & Fox, Brendan [5]	Modelling and simulation of underfloor heating system supplied from heat pump	The study highlights that underfloor heating systems supplied by heat pumps are more efficient than conventional radiator-based systems due to the larger heat emitting area and lower operational water temperature. This combination provides uniform floor temperature, better overall efficiency, and utilizes a significant portion of heat from renewable sources like outside air or ground instead of a fuel source, making it an optimal solution for heating.
Salam, Md Abdul and Islam, Md Mafizul [6]	Modelling and Control System design to control Water temperature in Heat Pump	Development of a combined PID and Model Predictive Controller (MPC) for an air-to-water heat pump system supplying domestic hot water, aiming to replace the existing PLC-based system due to its large size and high

Authors	Title	Main findings
		maintenance costs. The study found that the combined PID and MPC controller provided superior control performance, achieving minimal overshoot, rise time, and settling time compared to the PID alone, effectively maintaining the desired water temperature.
Ahmad, Muhammad Waseem [7]	Advanced control strategies for optimal operation of a combined solar and heat pump system	The study developed and compared conventional and advanced control strategies, including Model Predictive Controller (MPC), for a solar heating system combined with a heat pump in a two-zone building model. The findings indicate that MPC outperformed On-Off and PI controllers by providing better thermal comfort, reducing energy consumption, and optimizing the use of cheaper night-time electricity, thus lowering overall energy costs.
Byrne, Paul and Miriel, Jacques and L'enat, Yves [8]	Modelling and simulation of a heat pump for simultaneous heating and cooling	The study introduces a heat pump for simultaneous heating and cooling (HPS), which efficiently meets the heating, cooling, and hot water demands of small office and residential buildings, showing improved winter performance and defrosting capabilities compared to standard reversible heat pumps. Annual simulations using TRNSYS

Authors	Title	Main findings
		software reveal that the HPS can achieve up to 55% savings in electric energy consumption and a 19% improvement in annual performance.

Table 1 Literature Review

2.3 The Case Study

In the following sections, a comprehensive case study on the control of a heat pump system designed to regulate the temperature within a building is presented. The focus is on the development and implementation of an efficient control strategy to ensure optimal thermal comfort using a heat pump modeled in UniSim. The heat pump leverages the thermodynamic properties of a refrigerant, which undergoes various phase changes and pressure alterations to facilitate heating and cooling processes.

To maintain the desired temperature within the building, the speed of the compressor is controlled. This parameter is crucial as it directly influences the heating efficiency of the system. As a starting point, a detailed model of the heat pump system must be developed, it will then be possible to design and fine-tune a Proportional-Integral (PI) controller.

To integrate the control model with the actual heat pump system, an OPC DA (OLE for Process Control Data Access) interface is employed. This enabled communication between the controller interface and the physical system, allowing for the implementation of the developed control strategy in a live setting. This report details the modeling process in UniSim, the development and simulation of the control strategy in Simulink, and the practical implementation using OPC DA.

3 Heat pump model

3.1 Refrigeration cycle

A heat pump works on the principle of exchanging thermal energy opposite to the direction of spontaneous heat flow by absorbing heat from a cold space and releasing it to a warmer one. The whole process is composed by 4 steps explained below:

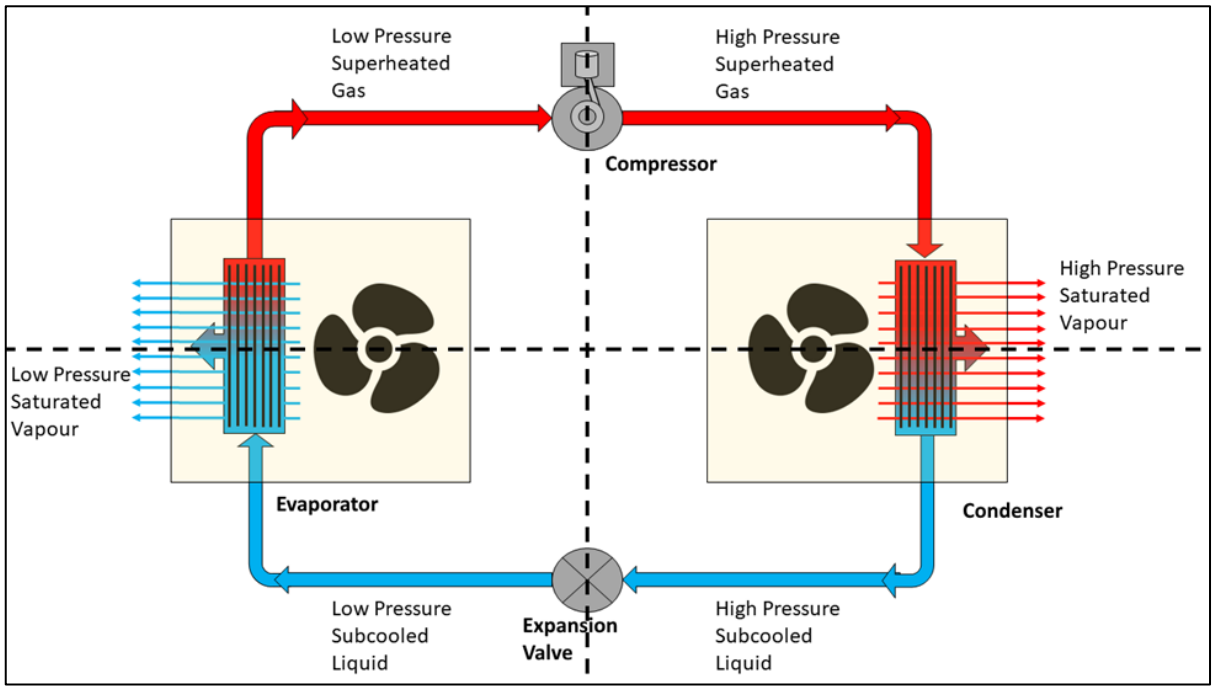


Figure 1 Heat Pump Refrigeration Cycle[1]

3.1.1 Compressor

The compressor is the element in charge of setting the refrigerant in motion and enabling it to flow through the entire system, as the refrigerant enters as a low-pressure, warm gas, it pressurizes it into high-pressure vapor. In line with the pressure increase, the temperature of the refrigerant rises substantially. This step is crucial for the next phase, as the refrigerant needs to be at a higher temperature than the outdoor air (in case of a heat pump in heating mode) or the heat rejection environment to effectively transfer heat. After compression, the refrigerant exits the compressor as a high-pressure, high-temperature vapor.

3.1.2 Condenser

Upon exiting the compressor as a high-pressure, heated gas, the refrigerant moves into the condenser. In this phase, the gaseous refrigerant undergoes a transformation into a liquid state while still maintaining high pressure, but with a temperature that's moderately warm. As it departs from the condenser, it retains this high-pressure liquid form but with a reduced warmth. The condenser in facilitating this temperature decrease allows the refrigerant to

[1] https://www.torr-engineering.com/the-refrigeration-cycle/

surpass its condensation threshold, guaranteeing its complete return to a liquid phase. This process of energy transfer not only enables the system to cool but can also be utilized for various applications, like heating a building's water supply in the given example. Hence, the refrigerant leaves this heat exchanger as a high pressure, subcooled liquid.

3.1.3 Expansion Valve

The refrigerant, now a high-pressure liquid at a standard temperature, is led into the expansion valve from the condenser. This valve is crucial for regulating the flow of refrigerant into the evaporator and is mechanically linked to the condenser via a diaphragm. A capillary tube from this diaphragm extends to a thermal bulb situated at the exit point of the evaporator. Located at the evaporator's discharge point, this thermal bulb is sensitive to the temperature of the emerging refrigerant. As the temperature at the evaporator's outflow varies, the refrigerant within the thermal bulb expands or contracts, causing corresponding pressure changes. These changes impact the capillary tube and, in turn, influence the diaphragm's positioning, dictating how much refrigerant is allowed to proceed to the next phase. Through this mechanism, the refrigerant is depressurized sufficiently, lowering its temperature beneath that of the surrounding air at the point of contact in the tubing, effectively shifting the refrigerant from its liquid state to a gaseous state.

3.1.4 Evaporator

Following its passage through the expansion valve, the refrigerant arrives in the evaporator as a low temperature, low-pressure liquid with vapor mixture. Here, it engages in another crucial heat exchange process, this time with the external air that needs cooling. The refrigerant starts to boil immediately after exiting the expansion valve, a process that persists until it has absorbed sufficient heat to transition fully from a liquid to a vapor state. Throughout the boiling phase, the refrigerant's temperature remains steady, a phenomenon known as saturation or evaporator temperature. Once the refrigerant completes its conversion to vapor, any further temperature increase is called superheat.

The term ‘superheat’ is used to refer to the quantity of heat that is added to a refrigerant vapor above its boiling point. This specific increase is vital to ensure that the refrigerant re-enters the compressor entirely in a gaseous state, without any liquid. As the compressor cannot compress liquid, any liquid ingress into it can damage the compressor. Hence, the refrigerant enters the compressor as a low pressure, superheated gas and the cycle repeats.

3.2 UniSim Model

The software used to represent the descripted model is UniSim, the model consists of four main components: A compressor, an expansion valve, and 2 heat exchangers that work as both condenser and evaporator; the condenser is modelled as a heat exchanger with water

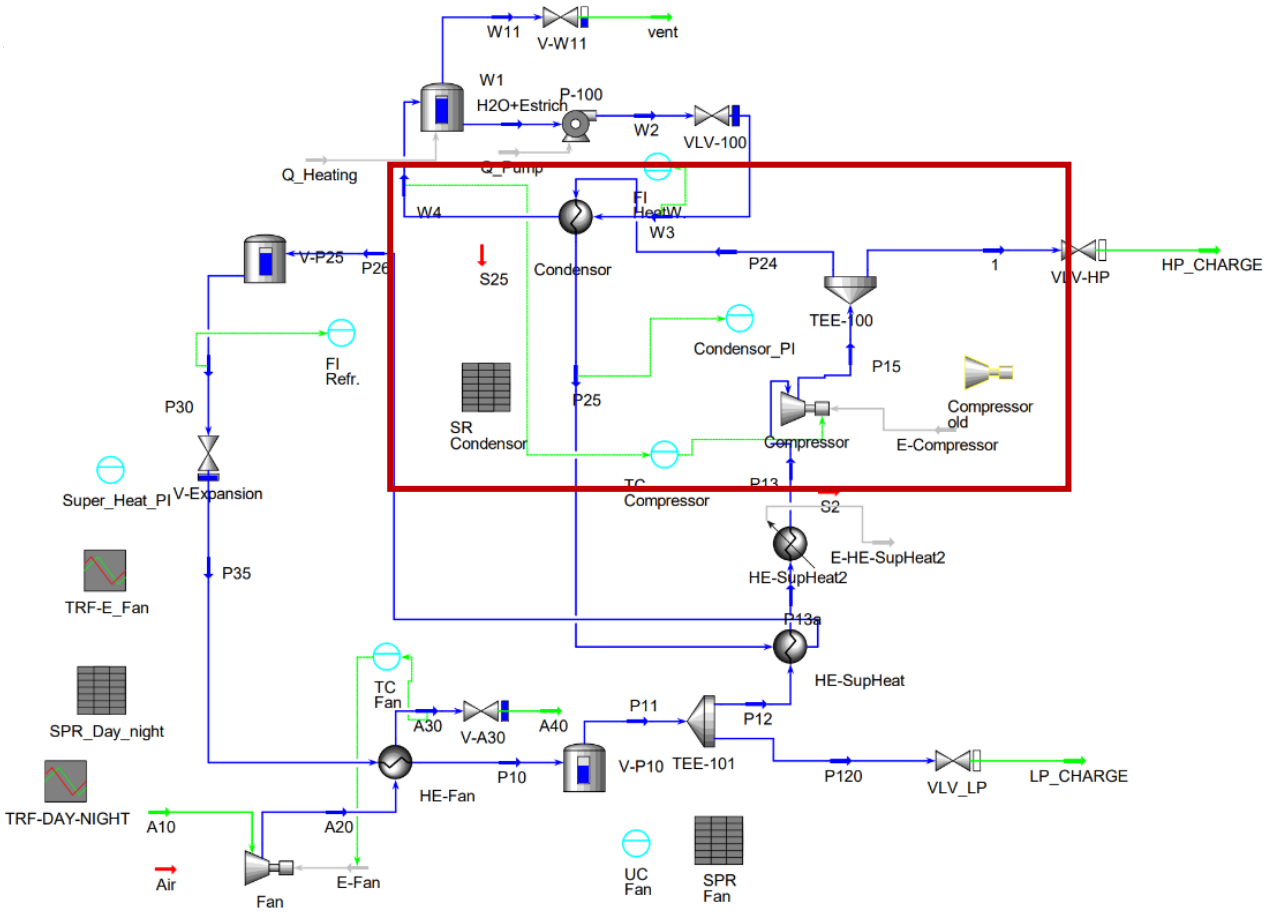


Figure 2. Heat pump model in Unisim

3.2.1 Control loops within the system

The refrigerant system at hand operates under the control of 2 individual loops, working independently. In the first control loop, temperature readings are collected from stream W4, which represents the flow of water within the building. This data then serves as the Process Variable (PV) and is communicated to the "TC-Compressor" PI controller. Based on this controller's specific settings, it determines an appropriate compressor speed, which is then implemented as either the Output (OP) or the Manipulated Variable (MV). Simply put, the internal water temperature dictates modifications to control the compressor's speed to achieve and maintain

a predetermined setpoint (SP). Adjusting the speed of the compressor has a direct impact on both the pressure and temperature of the refrigerant within the condenser section. Such alterations play a crucial role in regulating the heat transfer rate, thereby directly influencing the overall efficiency of the system.

In the second control loop, temperature readings taken from point A30 (where air has passed through the evaporator) are utilized as the Process Variable (PV). These readings are forwarded to the "TC-Fan" controller, which, in this context, is tasked with adjusting the speed of the fan responsible for introducing external air into the system. Here, the fan's speed becomes the Manipulated Variable (MV), a critical factor in regulating the system's performance.

Furthermore, the temperature of the outside air, simulated by stream A10, plays a significant role in this control loop. The external temperature affects the efficiency of heat transfer within the evaporator, necessitating its consideration in the system's operational strategy. This simulation approach enables the replication of varying conditions, such as day and night scenarios, allowing for a dynamic response to changes in external air temperature. Consequently, the desired internal water temperature within the building is subject to adjustments over time to accommodate these fluctuations. It's imperative to acknowledge that variations in outside air temperature introduce a degree of disturbance to the system's model. Hence, when designing a controller for this system, accounting for these external temperature changes is crucial to ensure optimal performance and adaptability to environmental conditions. This consideration ensures that the system can maintain desired internal conditions efficiently, despite external temperature fluctuations.

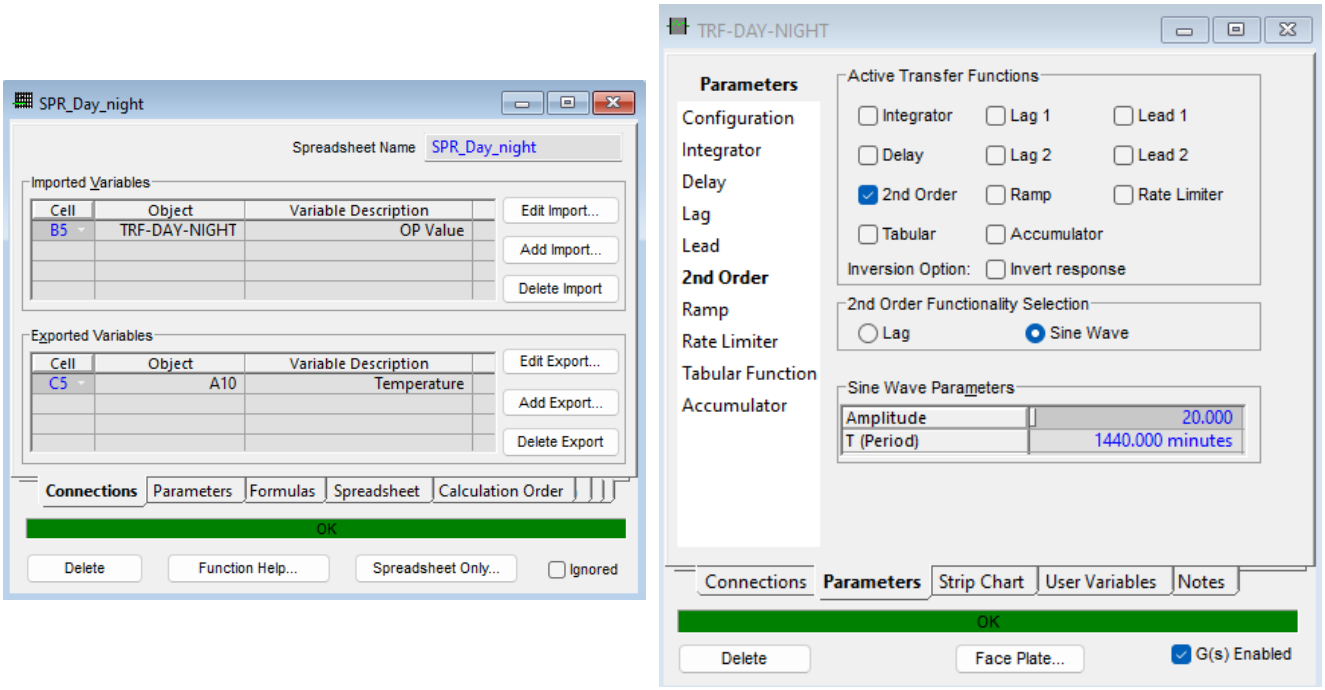


Figure 3. Day/Night Temperature Simulation using a Sinewave.

3.3 PRBS tests

The controller design will concentrate on the first control loop, where the compressor's revolutions per minute (RPM) are adjusted based on the water temperature within the building, this is the plant that will have to be modeled to design an external controller. To accurately identify the model's behavior and subsequently tailor the PI controller's settings, a technique known as Pseudo-Random Binary Sequence (PRBS) testing is employed. PRBS testing is instrumental in system identification — a critical step in controller design that involves developing a mathematical model of the system's response to input signals. By applying a PRBS signal, which alternates between predefined levels (in this case, varying the compressor speed by ±5% approximately, around a steady operating point).

The PRBS tests are conducted under different operating conditions, including variations in the time of day and corresponding outdoor temperatures, to observe the water temperature's response to changes in compressor speed. With this approach it is possible to gather data on how the system behaves under a wide range of conditions, reflecting the real-world operational variability a heat pump would experience. By analyzing how the building's water temperature, the process variable (PV), reacts to these modulations in compressor speed across different environmental conditions, it is possible to construct a detailed model of the system's dynamic behavior.

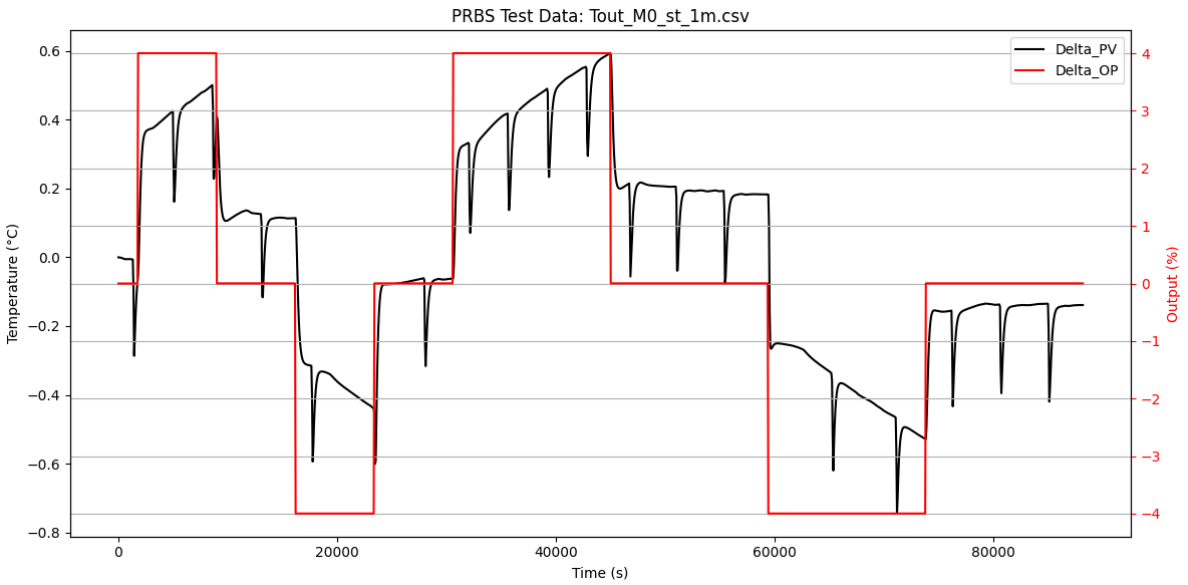
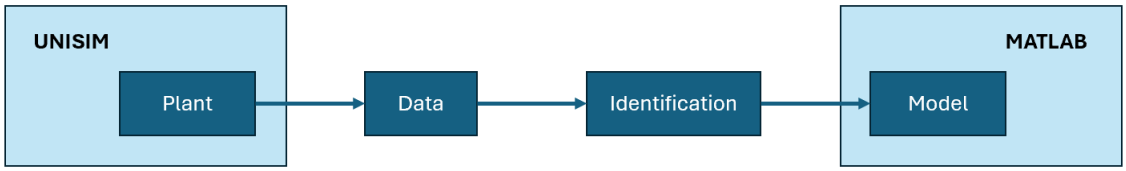


Figure 4 PRBS Test at -10°C

Utilizing PRBS tests in control systems is highly valued for several reasons. First, the robust nature of PRBS signals, with their high spectral density, ensures that the system is thoroughly tested across a wide range of frequencies. This is crucial for identifying the dynamic characteristics of complex systems. Additionally, PRBS testing helps in distinguishing between the system's true response and any background noise, which is essential for accurate system identification. Consequently, through these PRBS tests, a PI controller that adeptly manages the heat pump's compressor speed, optimizing the temperature control of the building’s water and enhancing the heat pump system's overall efficiency can be designed.

4 System Identification

System identification is an important step in our case study, involving the development of a mathematical model that represents the behavior of our Unisim model based on the observed data. There are many methods such as Impulse response identification, step response identification or PRBS signals for system identification.



For our model identification we have used Pseudo-Random Binary Sequence (PRBS) tests as mentioned in the previous section. We performed the tests at different Output Temperature (T_{Out}) as shown in the following table and switches up and down around its stationary point (e.g. At T_{Out} -11°C, stationary OP is 76.69 so the test is performed at +4 and -4 OP i.e. 80.69 and 72.69) for sufficient duration to capture system's dynamic response. Data is sampled at 1 minutes to capture the system dynamics accurately.

Table 2 Stationary Points of PRBS Tests

T_Out (°C)	**PV_Stationary** (°C)	**OP_Stationary** (%)
-12	33.902	97.0368
-11	33.7435	76.6963
-10	33.59	62.64
-9	33.4265	52.4591
-8	33.26	44.62
-5	32.62	30.45
-2	32.5	22.05
-1	32.17	19.96
0	31.92	18
1	31.84	16
2	31.69	14.45

4.1 Parameter Estimation: Gain (V) and Time Constants (T_i, T_1):

Parameter estimation is the process of computing the model's parameter values (V, T_i and T_1) from measured data. Accurate parameter estimation allows for the precise development of the model. In our case we focus on estimating parameters V, T_i , and T_1 using data obtained through Pseudo-Random Binary Sequence (PRBS) tests.

There are several methods for parameter estimation, each for its own application and use case including MATLAB Perimeter estimation toolbox as well as Python libraries like Gekko and SysIdentPy.

For our heat pump model, we have done the parameter estimation in MATLAB using the linear delta model to represent the system. During PRBS tests, we also identified the behavior of the system. Our control loop in Heat Pump model acts as an integrator and first order lag function in parallel so we have modeled that in Simulink which is mathematically represented as:

$$PV_{\Delta} = \left[\frac{1}{T_i s} + \left(\frac{V}{T_1 s + 1}\right)\right] \cdot (OP_{\Delta}) \quad (1)$$

where,

PV_{Δ} is the change in process variable
OP_{Δ} is the change in control output
T_i is the integral time constant
T_1 is the lag time constant
V is the gain

4.1.1 Modeling in Simulink:

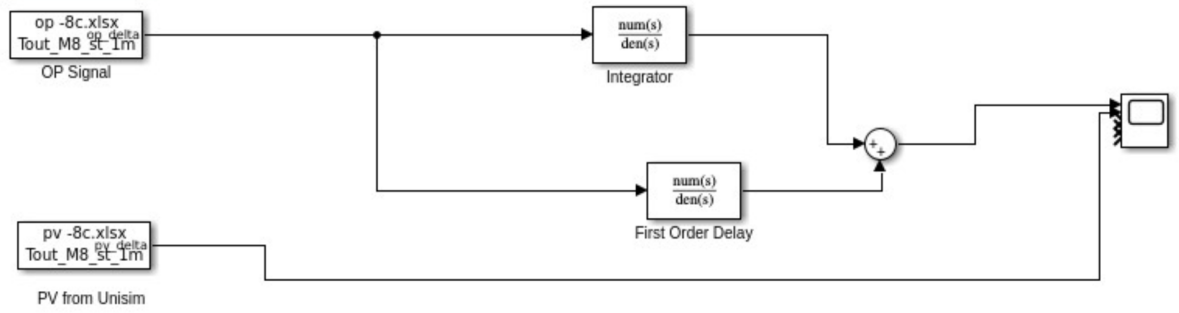


Figure 5 Linear Model in Simulink for Parameter Estimation

4.1.2 Comparing Model Responses

For the accurate parameter estimation, we compare the PV of Simulink model with the PV data obtained from UniSim at stationary OP. By adjusting the model parameter values (V, T_i, T_1) in the Simulink model, we tuned the model to match the response observed in UniSim.

This tuning process involves iterative adjustments and comparisons, ensuring that the simulated response closely aligns with the real system's behavior. The goal is to achieve minimal difference between the simulated and actual process variable responses as shown in the following figure, where blue line shows PV from UniSim model and purple line shows PV from Simulink model.

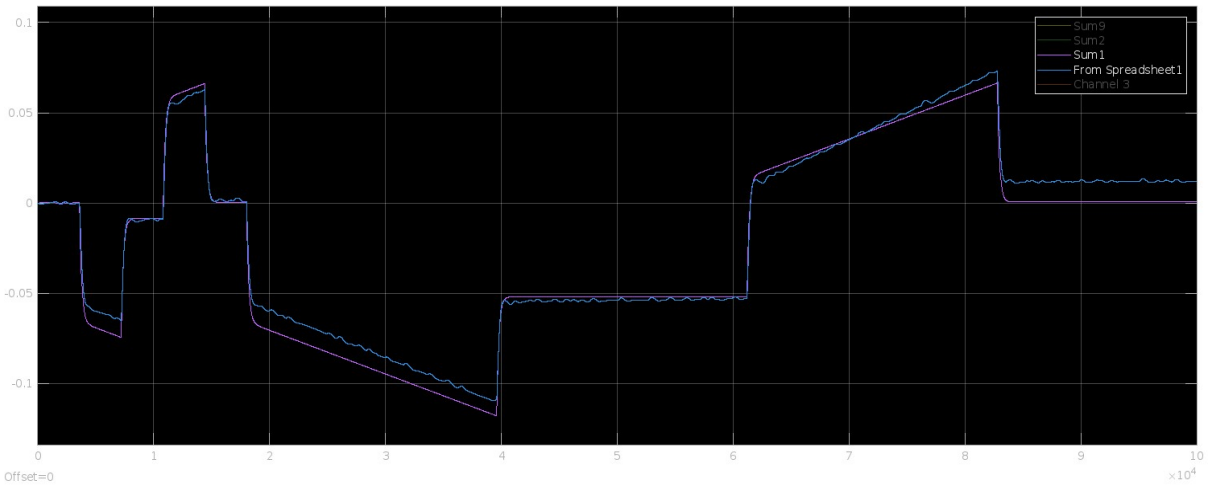


Figure 6 PV from UniSim and Simulink at Tout = -10°C

This parameter estimation is done at different T_{out} and stationary point and the results are summarized in the following table.

Table 3 Parameter Estimation Summary

T_Out (°C)	**PV_Stationary (°C)**	**OP_Stationary (%)**	**V**	**T_1 (s)**	**T_i (s)**	**T_i (days)**
-12	33.902	97.0368	0.01	9784.27	4025452	46.59
-11	33.7435	76.6963	0.01	242.77	1838599	21.28
-10	33.59	62.64	0.0165	180	1650000	19.09722
-9	33.4265	52.4591	0.019	250.49	1073817	12.42844
-8	33.26	44.62	0.025	250	949536	10.99
-5	32.62	30.45	0.06	300	550000	6.365741
-2	32.5	22.05	0.08	300	400000	4.62963
-1	32.17	19.96	0.08	300	220000	2.546296
0	31.92	18	0.13	300	300000	3.472222
1	31.84	16	0.11	300	280000	3.240741
2	31.69	14.45	0.1	300	250000	2.893519

From the analysis of the above estimation, we can see that the model parameter T_1 is constant for the changing OP but the parameters (V, T_i) are changing with respect to different control output OP. To address this, we can model these parameters as functions of the control output, using polynomial expressions.

4.1.3 Polynomial Representation of Parameters

To capture the dependency of parameters on the control output (OP), we represent each parameter as a polynomial function of OP. This approach allows us to describe how the parameters vary smoothly with changes in the operating point.

The general form of a polynomial function for a parameter p (where p could be V or T_i) is:

$$P(OP) = a_0 + a_1.OP + a_2 \cdot OP^2 + \cdots + a_n \cdot OP^n$$

(2)

Polynomial Regression for T_i: The relationship between Integrator time constant (T_i) and operating point (OP) is non-linear and can be represented by a second order polynomial given as:

$$T_i = 511.04\,OP^2 - 14301\,OP + 4053692$$

(3)

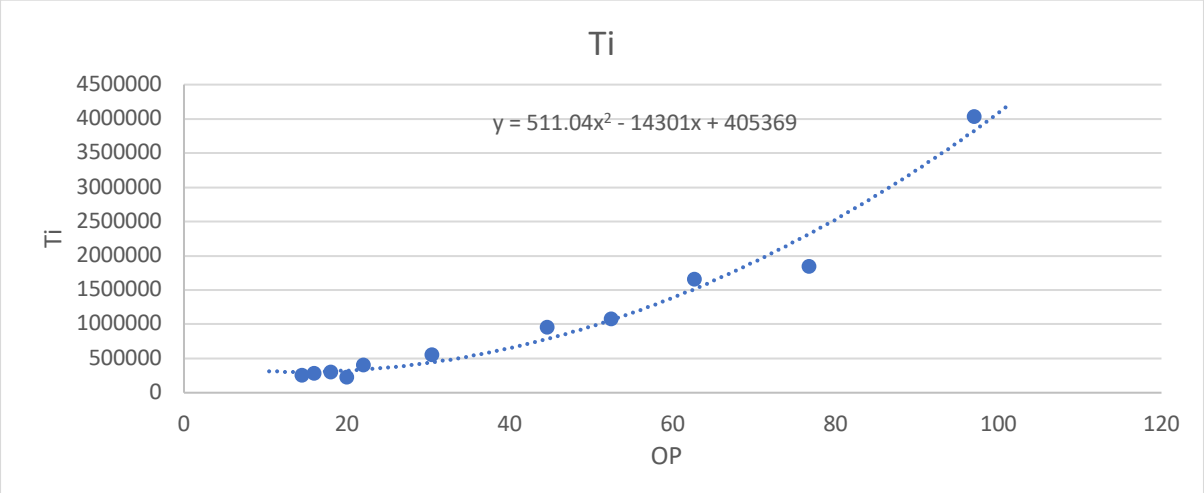


Figure 7 Curve Fitting for T_i

Polynomial Regression for V: The relationship between gain (V) and operating point (OP) is also non-linear and can be represented by a third order polynomial given as:

$$V = -3e^{-07}OP^3 + 3e^{-07}OP^2 - 0.0065OP + 0.1944$$

(4)

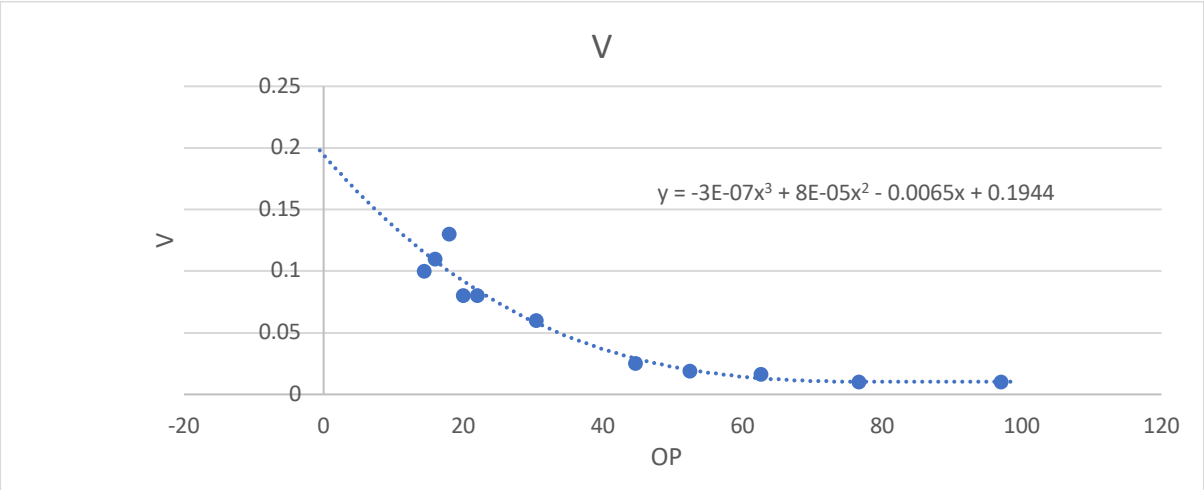


Figure 8 Curve Fitting for V

4.2 Using steady-state data to get polynomials

The next step in modeling is to understand how process variables (PV) and Output Temperature (T_{out}) relate to control output (OP) and setpoint (SP). By using steady state data, we can derive polynomial relationships that capture these dependencies by polynomial regression.

Non-Linear Stationary Curve: For the modeling of the system, it is very important for us to understand the steady state response of the system to different input values, capturing the relationship between the input and the output of the system. The Non-Linear Stationary Curve is represented by a second order polynomial below:

$$PV_S = -0.0004 \cdot (OP_S)^2 + 0.0693 \cdot (OP_S) + 30.896 \quad (5)$$

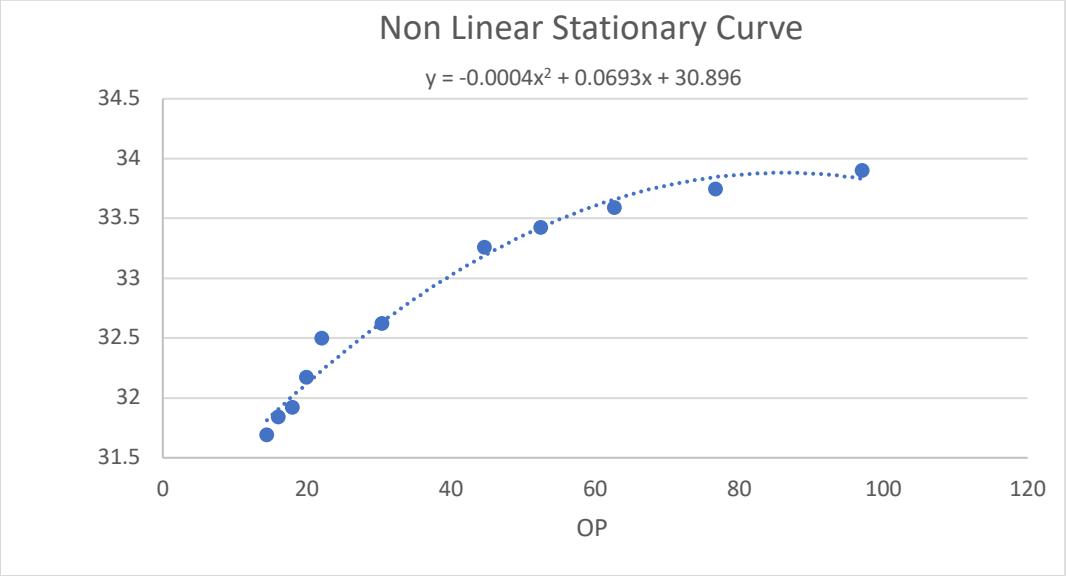


Figure 9 Non Linear Stationary Curve

Effect of Disturbance T_{out}: To understand the effect of disturbance (atmospheric temperature T_{out}) on our system, we need to find the relationship between OP and T_{out}. This analysis helps in identifying how external temperature fluctuations influence the system's performance, allowing for the design of robust control strategies that can mitigate or compensate for these disturbances. The effect of T_{out} is represented by a second order polynomial given as:

$$OP = 0.5285 \cdot (T_{out})^2 + 0.2364 \cdot (T_{out}) + 16.302 \quad (6)$$

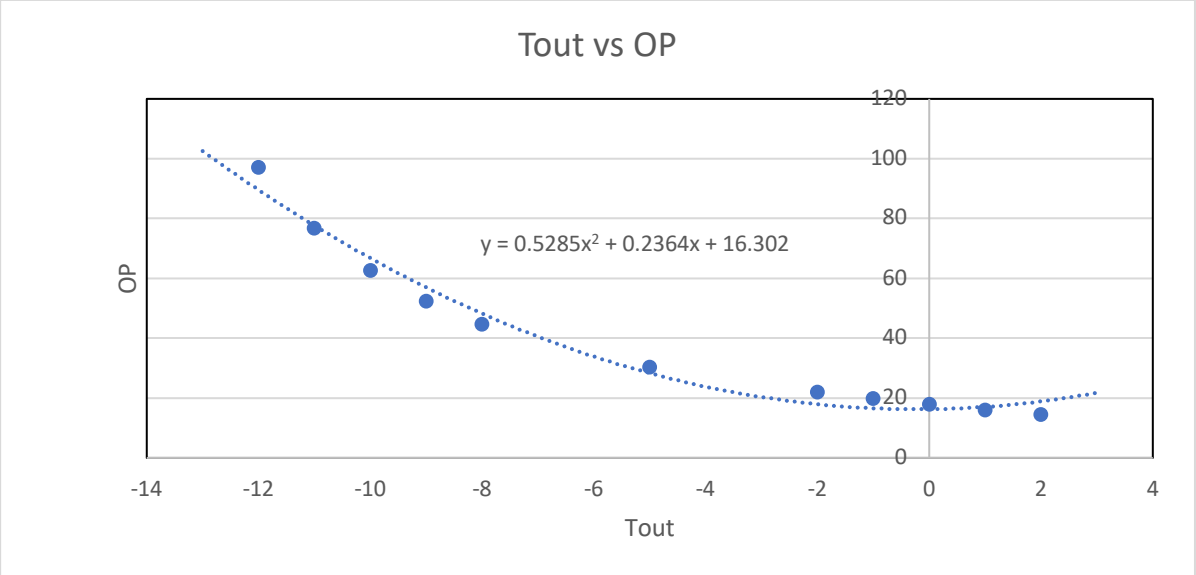


Figure 10 Effect of Disturbance(Tout) on OP

Setpoint Tracking: For setpoint tracking we also found the relationship between OP and PV given as:

$$OP_S = 20.075 \cdot (PV_S)^2 - 1284.7 \cdot (PV_S) + 20570$$

(7)

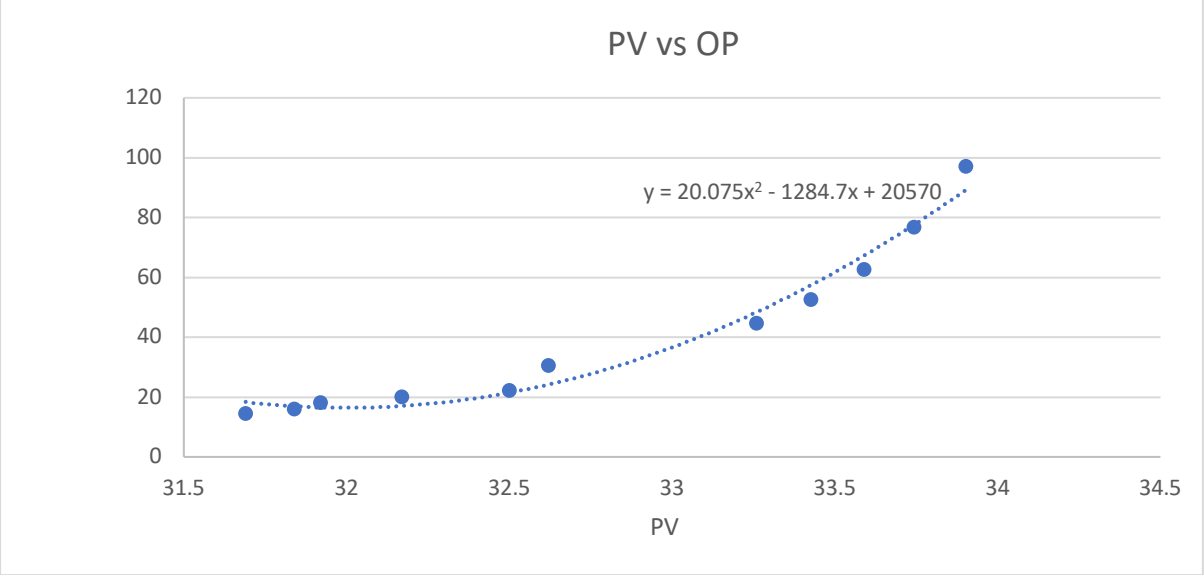


Figure 11 Setpoint Tracking Curve

Effect of Disturbance T_{out} on Setpoint: The effect of disturbance T_{out} is linearly related with Setpoint and given as:

$$PV = -0.1576 \cdot (T_{out}) + 31.999$$

(8)

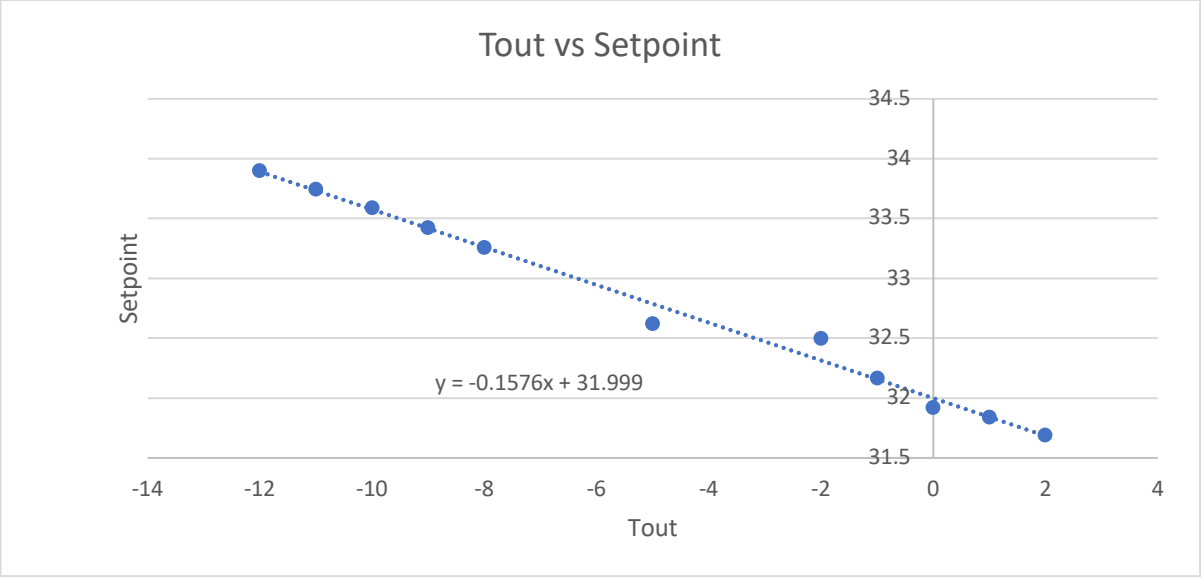


Figure 12 Effect of Disturbance on Setpoint

4.3 System Model for the Heat Pump

An additional part of the research work was an attempt to develop a model for the heat pump system (initially with the intention to be used as an internal plant for the model predictive controller and eventually used to test the PID controller). The entire system model was considered to be a combination of a stationary non-linear curve and the linear delta model at each stationary point. The non-linear polynomial describes the stable behavior of the system and the relationship between PV (temperature of H_2O) and the OP (compressor speed as percentage). Along this stationary curve, the linear delta model describes the linearized behavior at stable points.

The non-linear relationship was found out by fitting a curve between the stationary OP_S and PV_S values (Table 3) and is defined by the polynomial,

$$PV_S = -0.0004 \cdot (OP_S)^2 + 0.0693 \cdot (OP_S) + 30.896 \quad (9)$$

This relation helps the system move on the stationary curve depending on the stationary OP value at the given instance.

To represent the linear behavior of the system, an integrator and a lag in parallel was considered as the linear model, and is represented as,

$$PV_\Delta = \left[\frac{1}{T_i s} + \left(\frac{V}{T_1 s + 1}\right)\right] \cdot (OP_\Delta) \quad (10)$$

The system model variables T_i, T_1 and V vary depending on the stationary OP_S. Using the identified data these can be represented as polynomials as described in section 4.1.

Therefore, using $OP = OP_S + OP_\Delta$, the stable point and the linearized behavior can be used to compute $PV = PV_S + PV_\Delta$.

This is modelled in Simulink as,

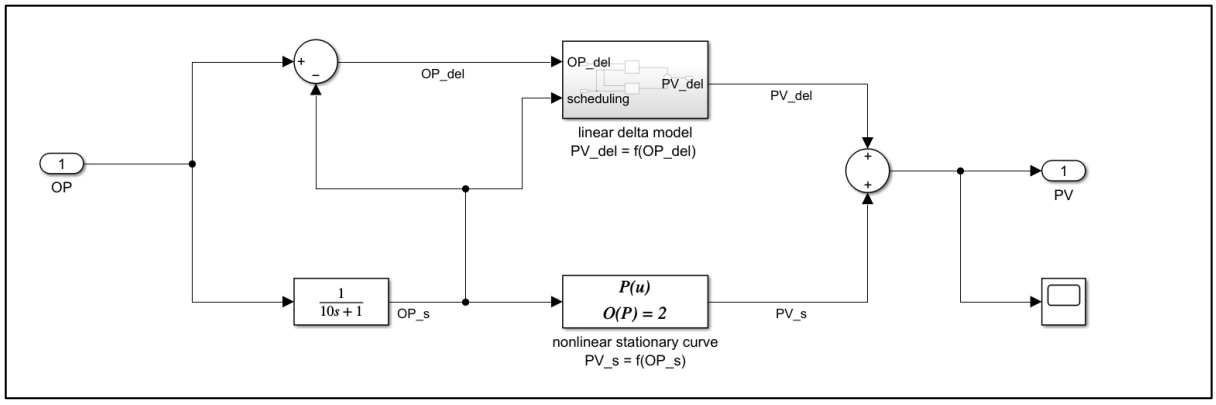


Figure 13 Heat Pump Simulink Model

As stated earlier, the Simulink model consists of a stationary and a delta model. To achieve OP_Δ in the model, a lag element was used to create the OP_S and eliminate it from OP. Then the PV_S and the PV_Δ obtained after simulating were summed up to calculate PV.

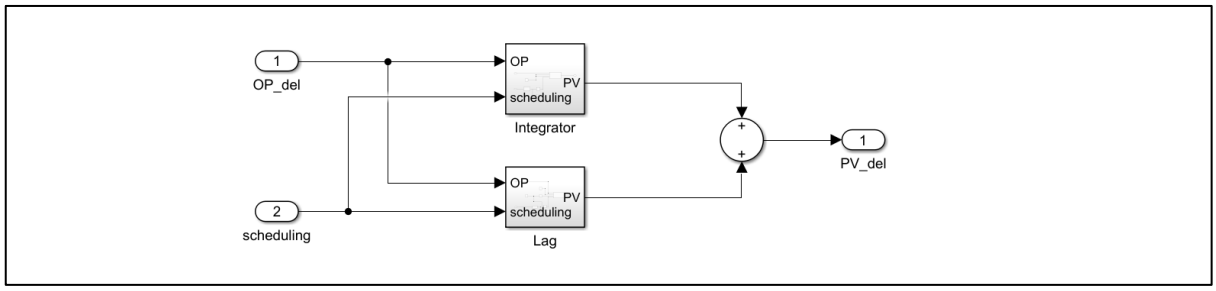


Figure 14 Linear delta model in Simulink

The linear delta model is made up of an integrator and a lag element in parallel. Additionally, the linear model also consists of a "scheduling" input port, which is used to determine the appropriate values for the system parameters T_i, T_1 and V depending on the stable point of the system (OP_S).

In this way, the heat pump was modelled in Simulink and was used for further testing.

4.4 System control

In order to control the system, PID control scheme was employed. To supplement the PID controller, feedforward control was also implemented and tested. Firstly, a feedforward loop to enhance the setpoint tracking of the controller was implemented using $OP = f(SP)$ [eqn. (7)]. This controller helps the system to reach the appropriate OP directly, depending on the desired PV value. This then only leaves the PID control to minimize the error around the stable point.

Additionally, in order to reject the effect of disturbance (atmospheric temperature T_{out}), another feedforward control was implemented using the relation $OP = f(T_{out})$ [eqn. (6)]. Also to have a more stable reaction and not act too quickly, a lag was used in the disturbance rejection loop.

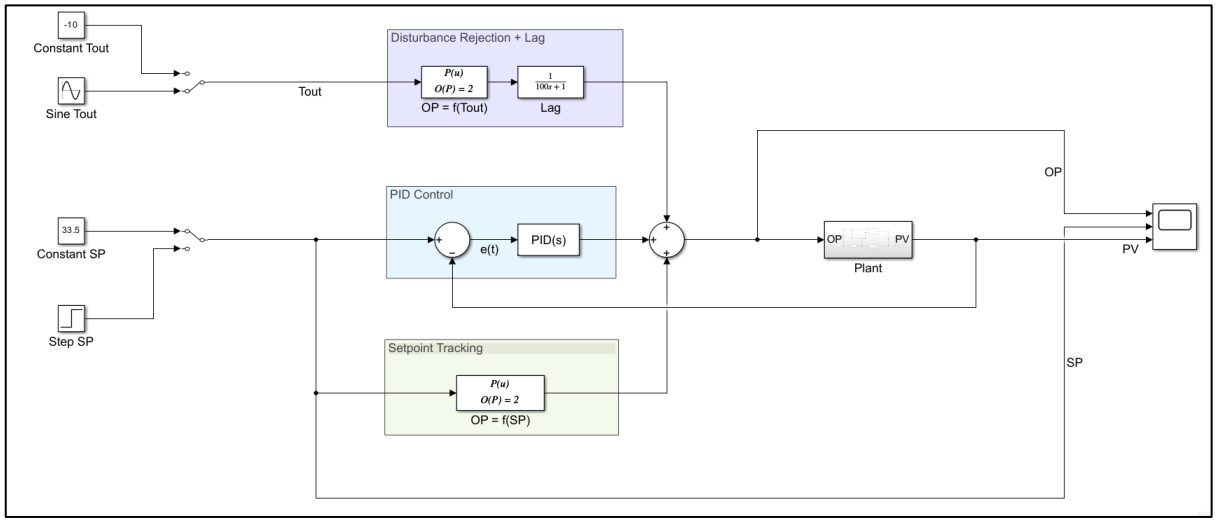


Figure 15. Simulation environment in Simulink

Figure 15 shows the complete simulation setup with the plant model, the PID controller, and the two feedforward controllers. To replicate the real scenario, where the atmospheric temperature varies during the 24 hours of the day, a sinusoidal wave was also used in the simulation. This was eventually replaced by the real T_{out} values from the actual plant (Unisim model).

5 OPC DA

5.1 Introduction

OPC DA (Open Platform Communications Data Access) is one of the industrial communication standards which allows real time data exchange between a server and a client. These servers and clients could be devices like control devices or software's from various manufactures. It's important to note that UniSim currently supports only OPC DA and not OPC UA (Unified Architecture).

Matrikon OPC is one of the solution providers which support OPC DA connectivity. Hence in the case study the MatrikonOPC Server for Simulation and Testing is the OPC Server which bridges the communication gap between UniSim and MATLAB/Simulink. Additionally, the MatrikonOPC Explorer is the OPC Client which is used to monitor the compressors OP & PV values, acting as a dashboard to provide real-time visualization of the heat pump system's performance.

5.2 How It Works

The integrated system operates by using OPC DA standards to enable real-time communication and control between Simulink and UniSim through the Matrikon OPC Server.

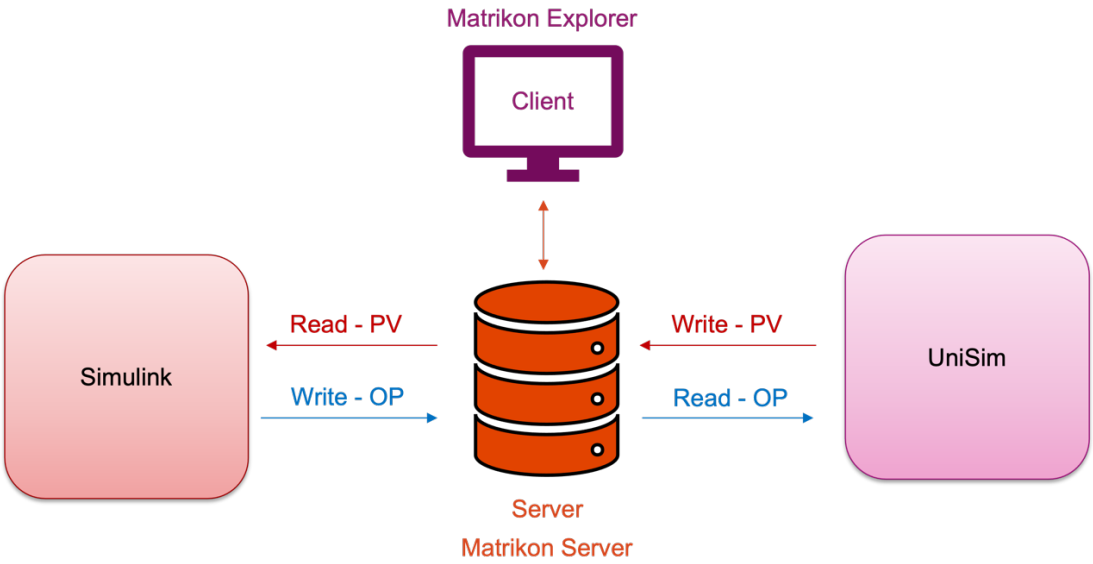


Figure 16 OPC - Data and Control Flow

5.2.1 Data Flow

- UniSim sends real time process data to the OPC Server.
- OPC Server makes this data available to connected clients.

5.2.2 Control Flow

- Simulink, as an OPC client, reads the process variable (PV) from the Matrikon OPC Server.
- Based on the control algorithm implemented in Simulink, control signals (Compressor's Operating Point (OP)) are generated.
- These control signals are sent back to the OPC Server, which forwards them to UniSim to adjust the compressor speed

5.3 Prerequisites

Before proceeding with the setup, ensure that the following prerequisites are met:

- UniSim: Installed and properly licensed.
- MATLAB/Simulink: Installed with a valid license, including the OPC Toolbox (Industrial Communication Toolbox)
- Matrikon OPC Server for Simulation and Testing - Server
- Matrikon OPC Explorer – Client

5.4 Setting up OPC Server

1. Open MatrikonOPC Server for Simulation and Testing which opens the configuration screen, as shown in *(*Figure 17*)*

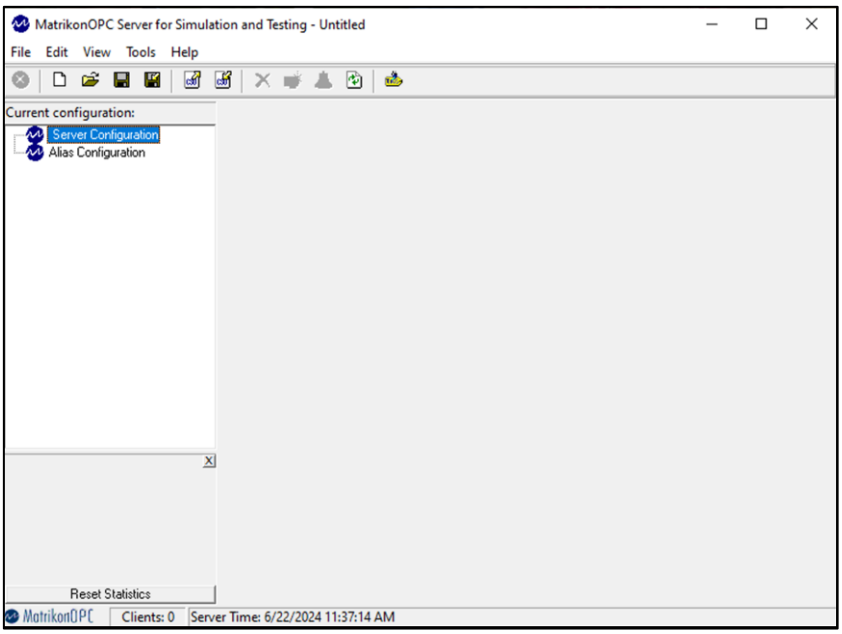


Figure 17 MatrikonOPC Server for Simulation and Testing - Configuration Screen

2. In the left side of the configuration screen, select "Alias Configuration" which then opens the contents pane. Right click on Alias Configuration and choose "Insert Alias Group" and give it a name Eg. CaseStudy, as shown in *(*Figure 18*)*

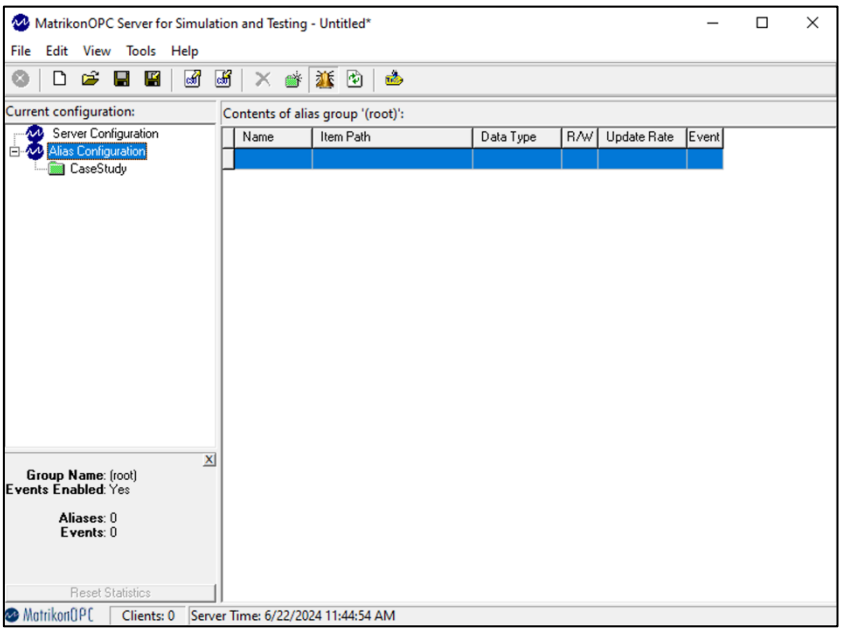


Figure 18 MatrikonOPC Server for Simulation and Testing - New Alias Group

3. In the contents pane, either right click and choose Insert New Alias or double click on the row to add a new alias, which opens the new alias window, as shown in *(*Figure 19*)*

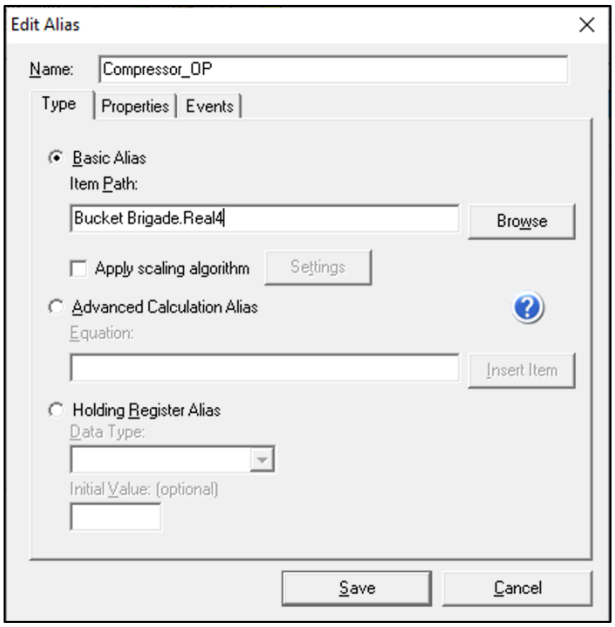


Figure 19 MatrikonOPC Server for Simulation and Testing - New Alias Window

4. Enter the name of the Tag, and choose an Item Path. Item Path is the path where the data is stored / retrieved. Hence make sure that no two Tag names use the same Item Path, unless required. Once done the newly created aliases should appear, as shown in *(*Figure 20*)*

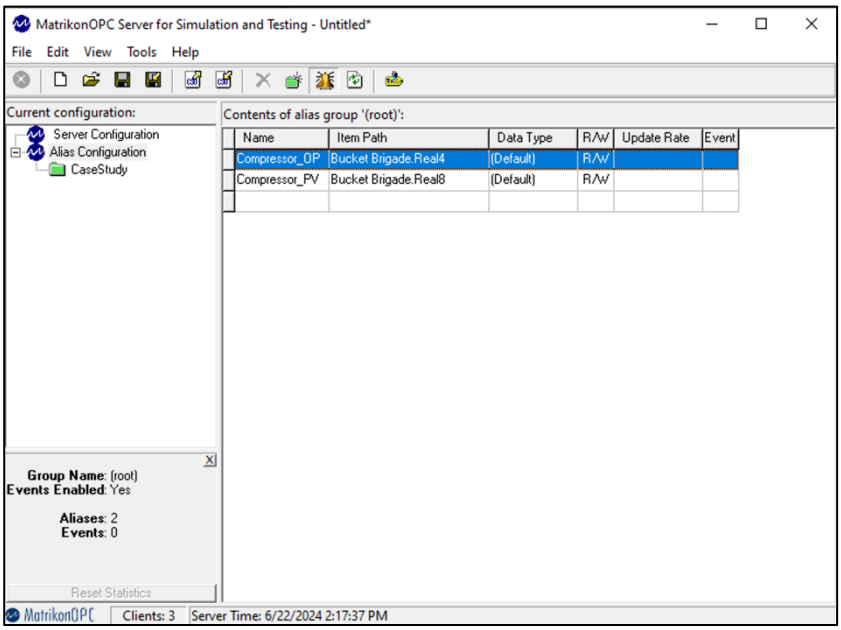


Figure 20 MatrikonOPC Server for Simulation and Testing - New Aliases

5.5 Setting up OPC Client

1. Open MatrikonOPC Explorer which opens the configuration screen, as shown *in (*Figure 21*).*

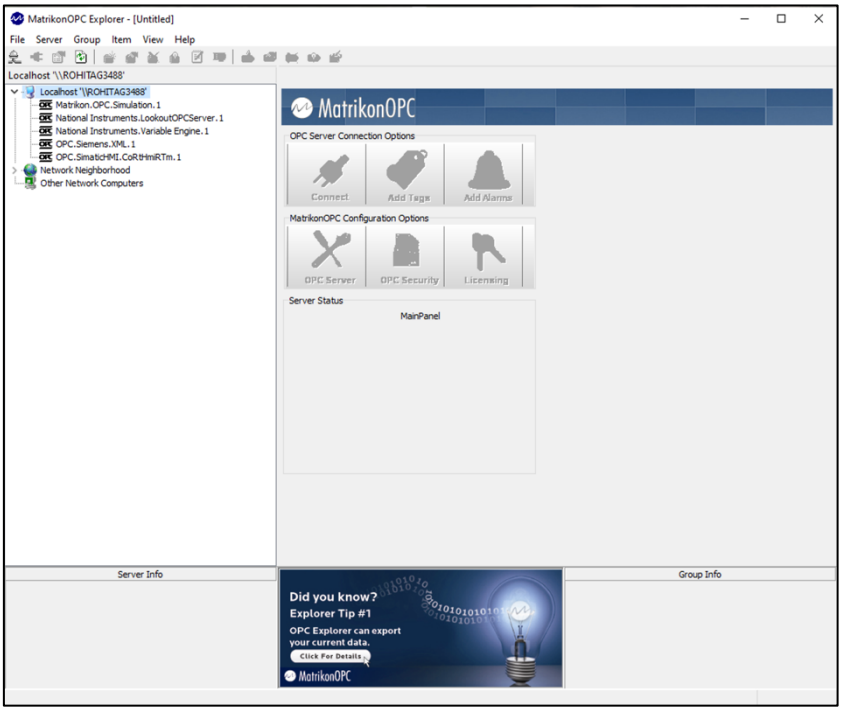


Figure 21 MatrikonOPC Explorer - Configuration Screen

2. On the right hand side of the configuration screen, you will be able to see the OPC Server which is running on the local computer. Make sure to choose the appropriate OPC Server and click connect, as shown in *(*Figure 22*)* and *(*Figure 23*).*

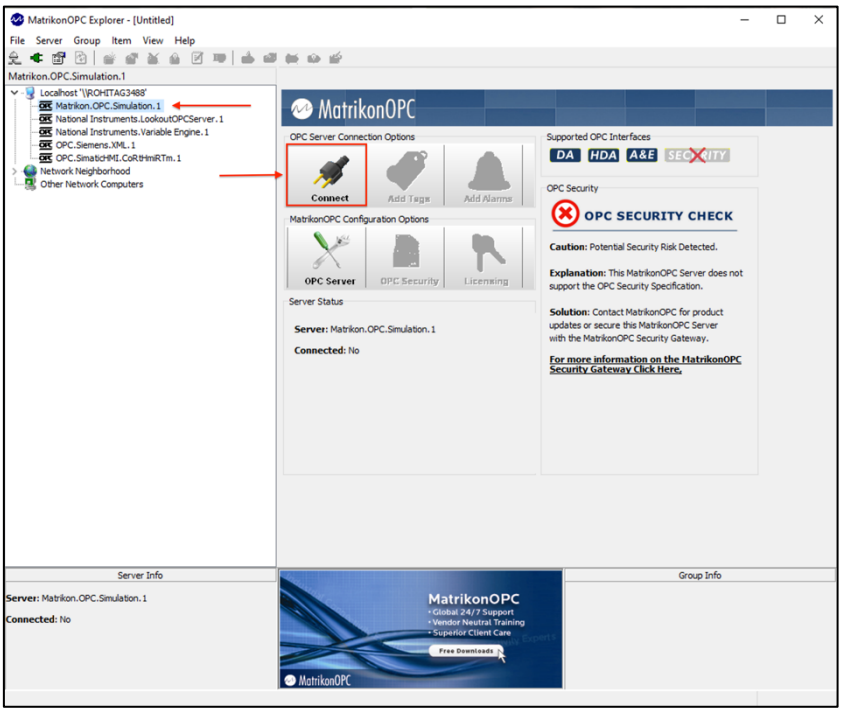


Figure 22 MatrikonOPC Explorer - Before Connecting

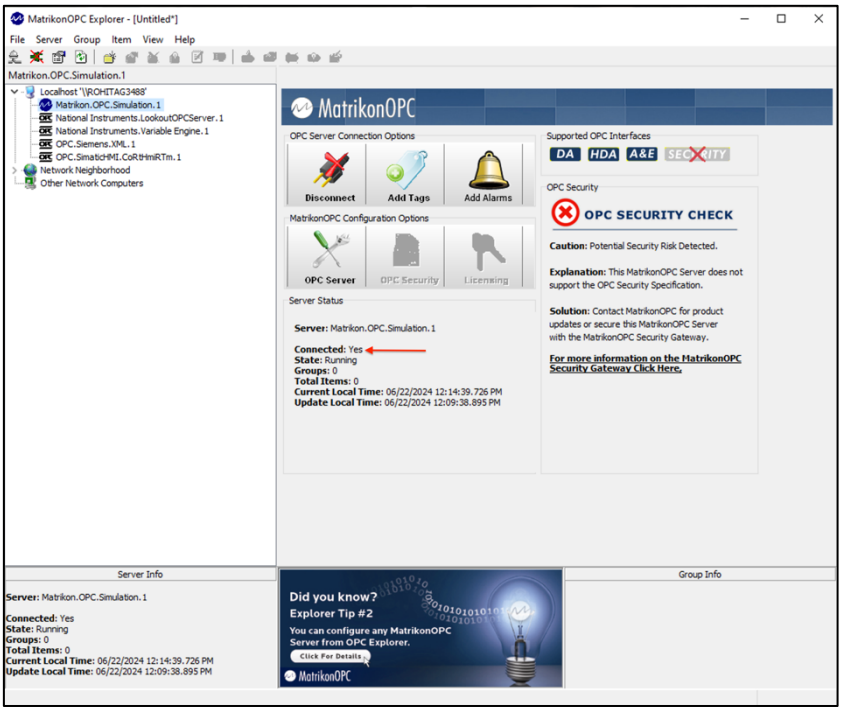


Figure 23 MatrikonOPC Explorer - After Connecting

3. In order to monitor the value on the MatrikonOPC Explorer, data points has to be added. Click on the Add Tag button in the configuration screen, as shown in *(*Figure 24*)*.

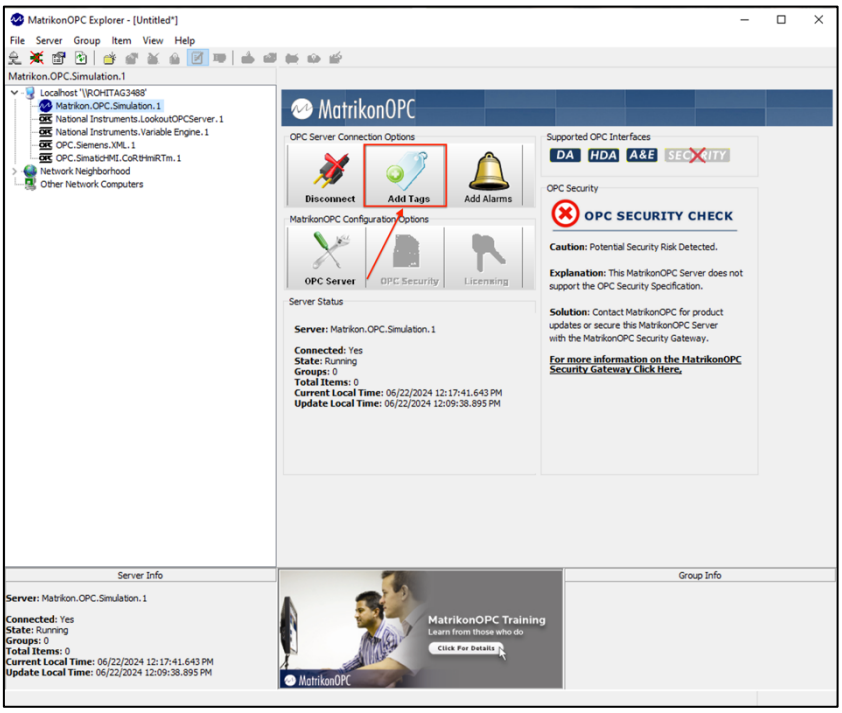


Figure 24 MatrikonOPC Explorer - Add Tags

4. In the browsing window, under Available items, choose the Alias name which was created as per *(Figure* 18*),* and the tags which was created is will be visible. Choose both the tags and select “Add Tags to list button”, as shown in *(*Figure 25*).*

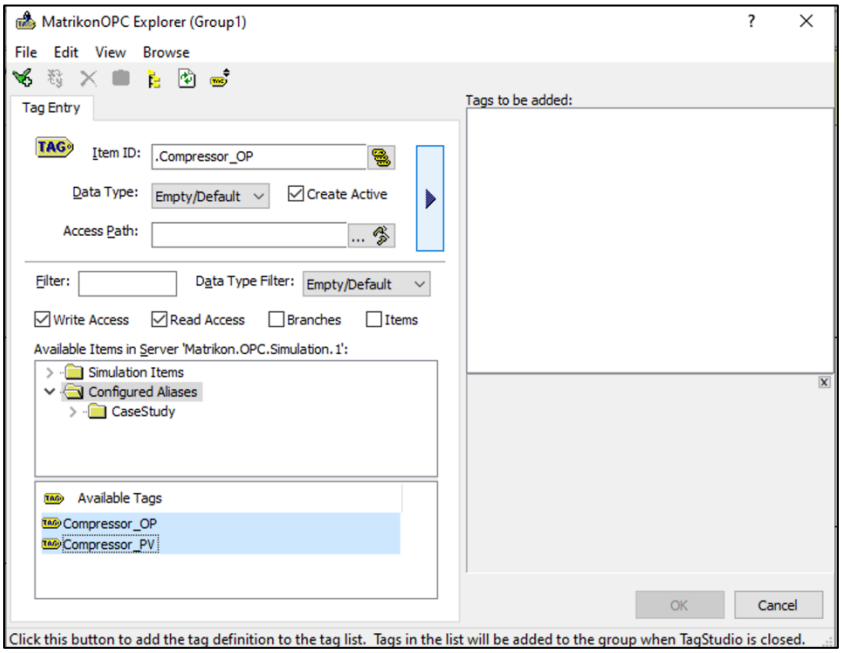


Figure 25 MatrikonOPC Explorer - Adding Tags to List

5. Once the tags are added, the configuration screen shows the tag name along with the quality and status of the tag weather it is active or not, as shown in *(*Figure 26*).*

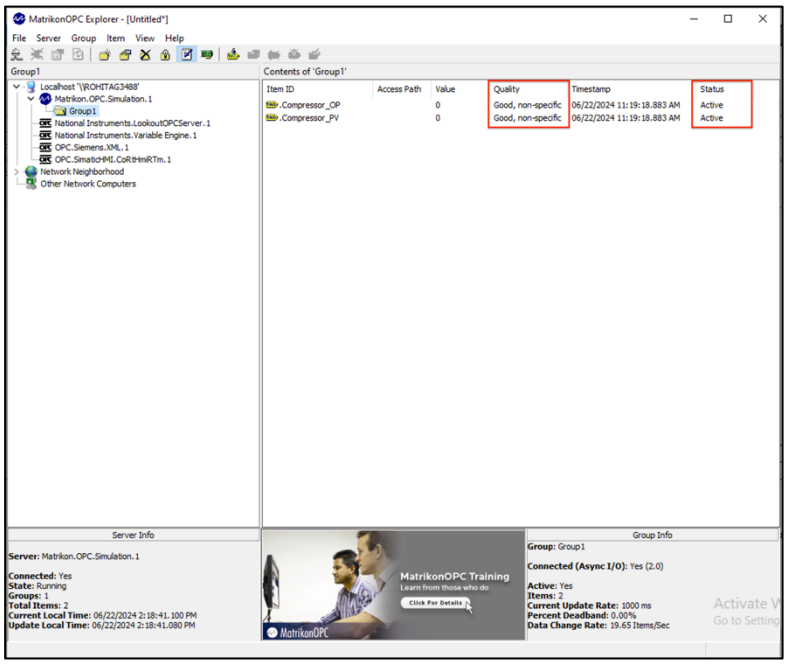


Figure 26 MatrikonOPC Explorer - Dashboard

5.6 Setting up UniSim with OPC Server

5.6.1 Adding variable which has to be monitored

6. Go to Tools -> DataBook. In the DataBook dialog box, go to the tab “Variable”. Click on Insert which open the “Variable Navigator” window. Choose the object who’s variable has to be added, Eg. TC Compressor OP & TC Compressor PV, as shown in *(*Figure 27*) and (*Figure 28*).*

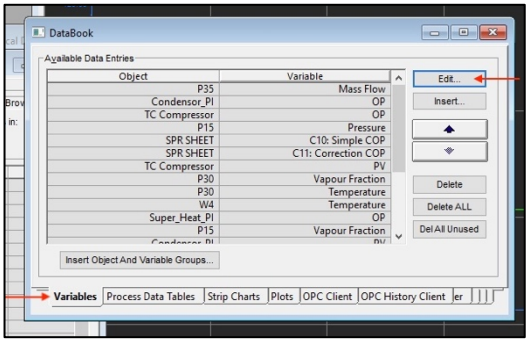


Figure 27 UniSim - DataBook Window - Variable Tab

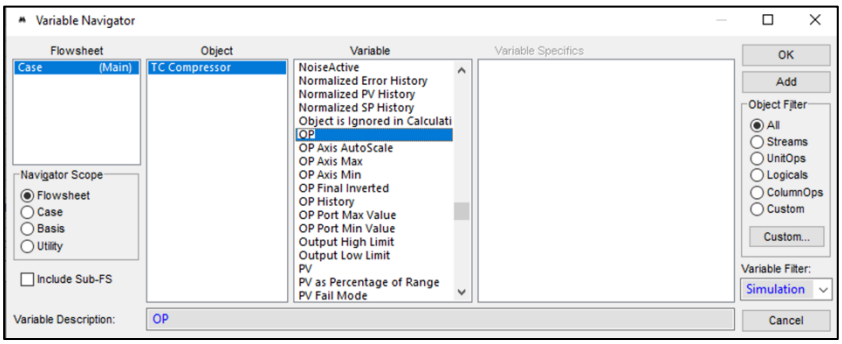


Figure 28 UniSim - Variable Navigator

5.6.2 Connecting Matrikon Server with UniSim

7. In the DataBook dialog box, go to the tab “OPC Client”. Choose the appropriate OPC Server and tick the check box “Tag Browser”, as shown in *(*Figure 29*)*.

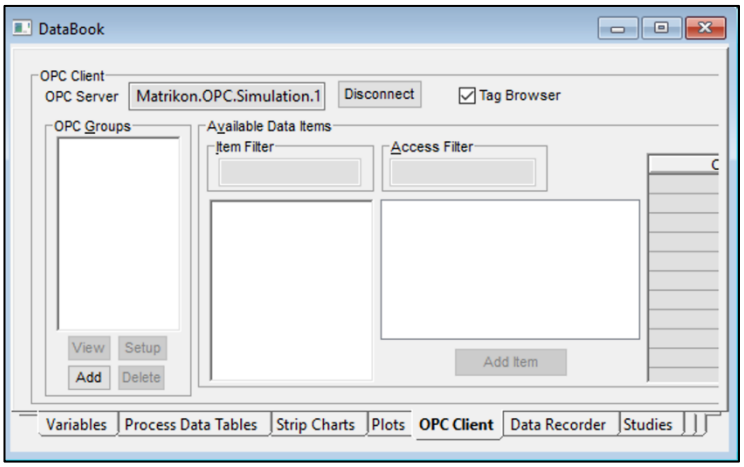


Figure 29 UniSim - DataBook - OPC Client Tab

8. Under OPC Groups, Click on add. A new group is created. Under “Configured Aliases”, the previously created tags will be visible. Select the Tag and click on “Add Item”, as shown in *(*Figure 30*)*.

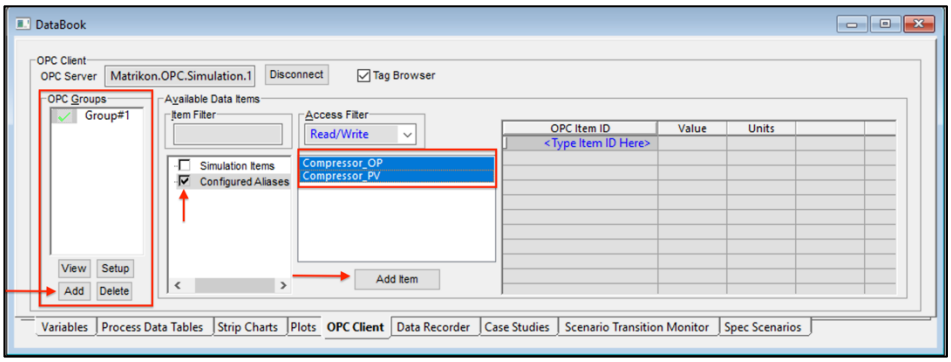


Figure 30 UniSim - Adding Tags

9. Under OPC Groups, Click on View, a dialog box opens where the Tags has to be linked with the Objects from the UniSim and their Variable, as shown in *(*Figure 31*)*.

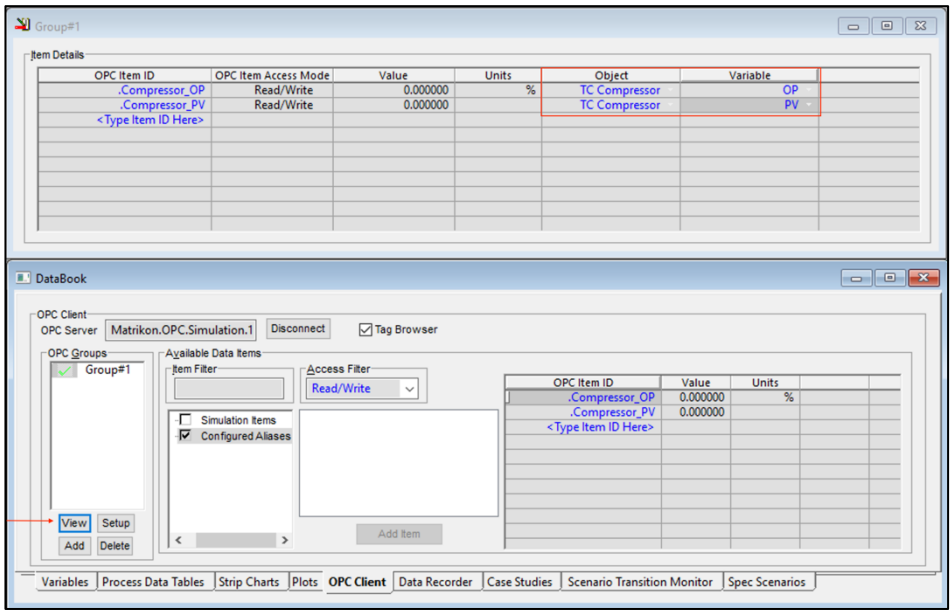


Figure 31 UniSim - Linking Tags and Object Variables

5.7 Setting up MATLAB/Simulink with Matrikon Server

10. Go to MATLAB -> Simulink. Once the model is opened in Simulink, go to “Library Brower” and search for the OPC Blocks (OPC Configuration, OPC Read and OPC Write, as show in *(*Figure 32*)*.

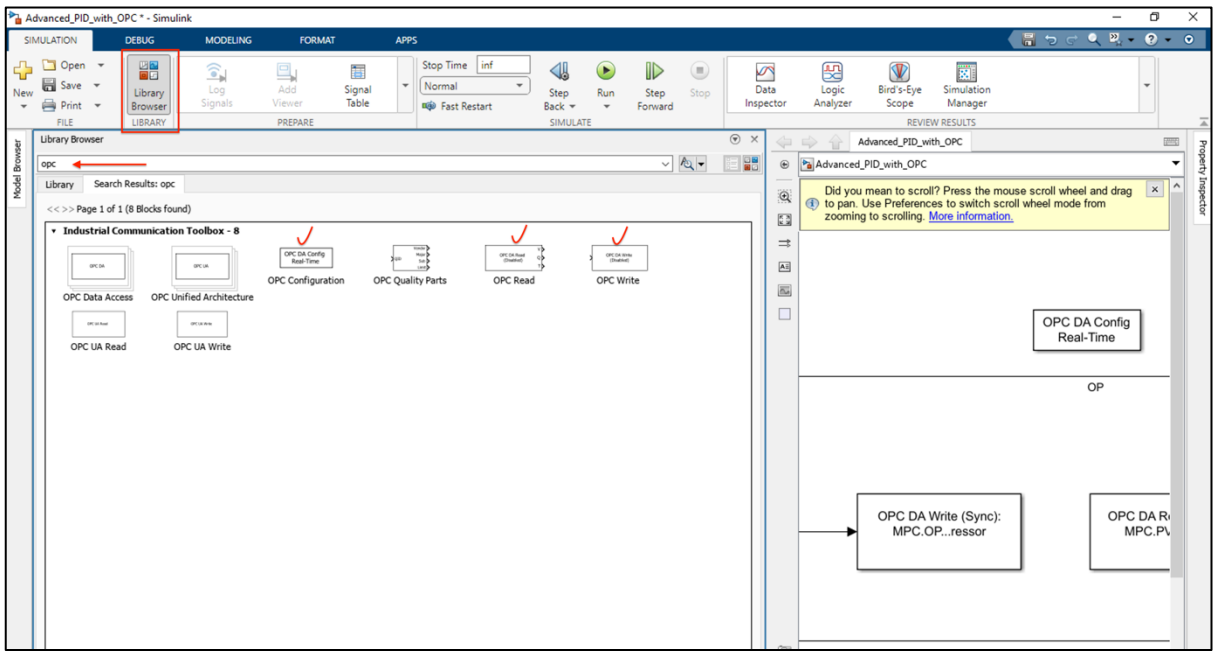


Figure 32 Simulink - Library Browser Window

11. Double click on the “OPC DA Config Real-Time” block -> Configure OPC Clients -> Add-> Select Server -> Click on OK, as shown in *(*Figure 33*)*.

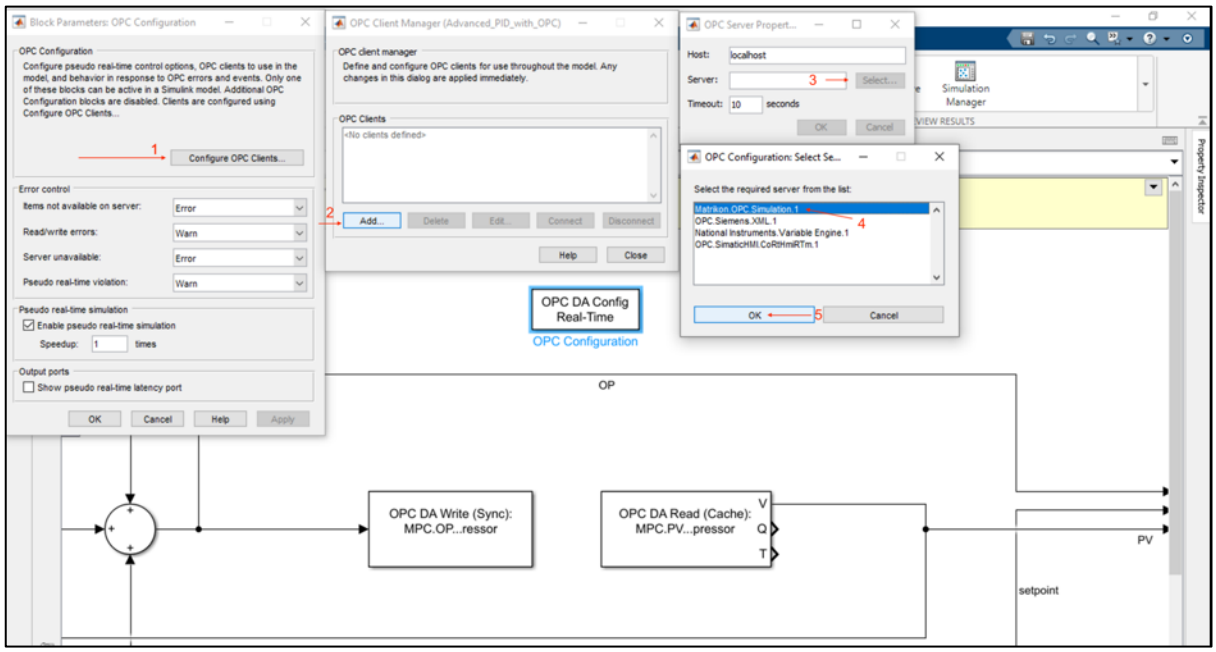


Figure 33 Simulink - OPC DA Configuration – Client

12. Double Click on the “OPC DA Write” block -> Add Items-> Configured Aliases -> Compressor_OP -> OK, as shown in *(*Figure 34*)*.

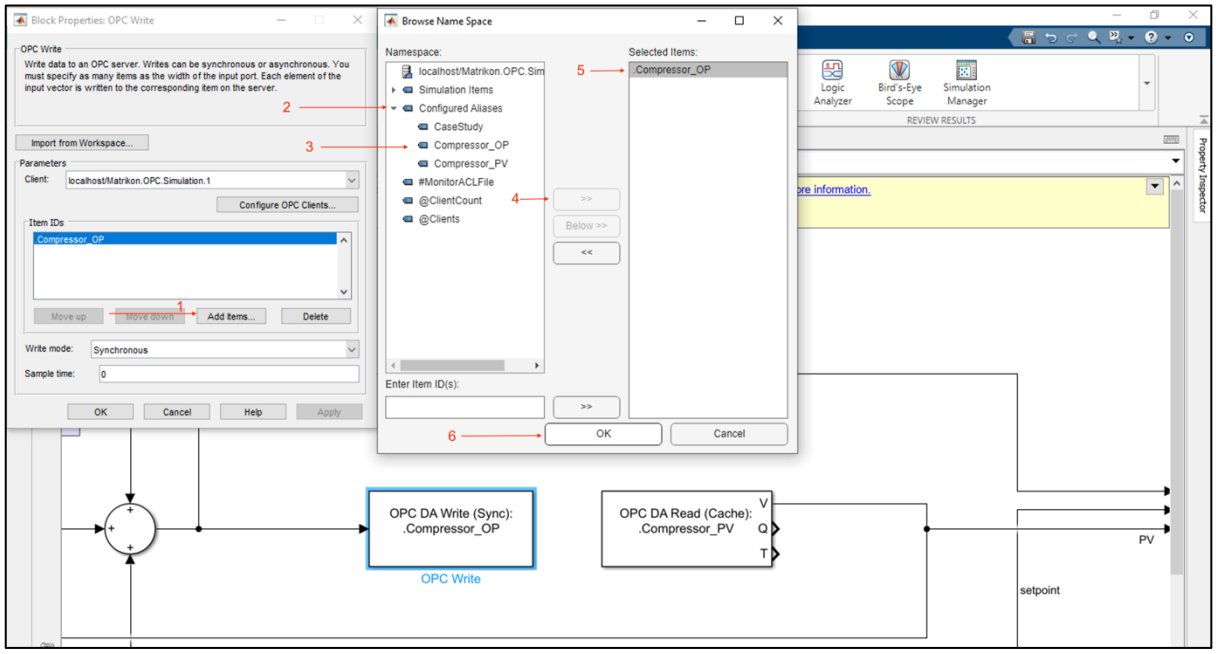


Figure 34 Simulink - OPC DA Write Configuration

13. Double Click on the “OPC DA Write” block -> Add Items-> Configured Aliases -> Compressor_PV -> OK, as shown in *(*Figure 35*)*.

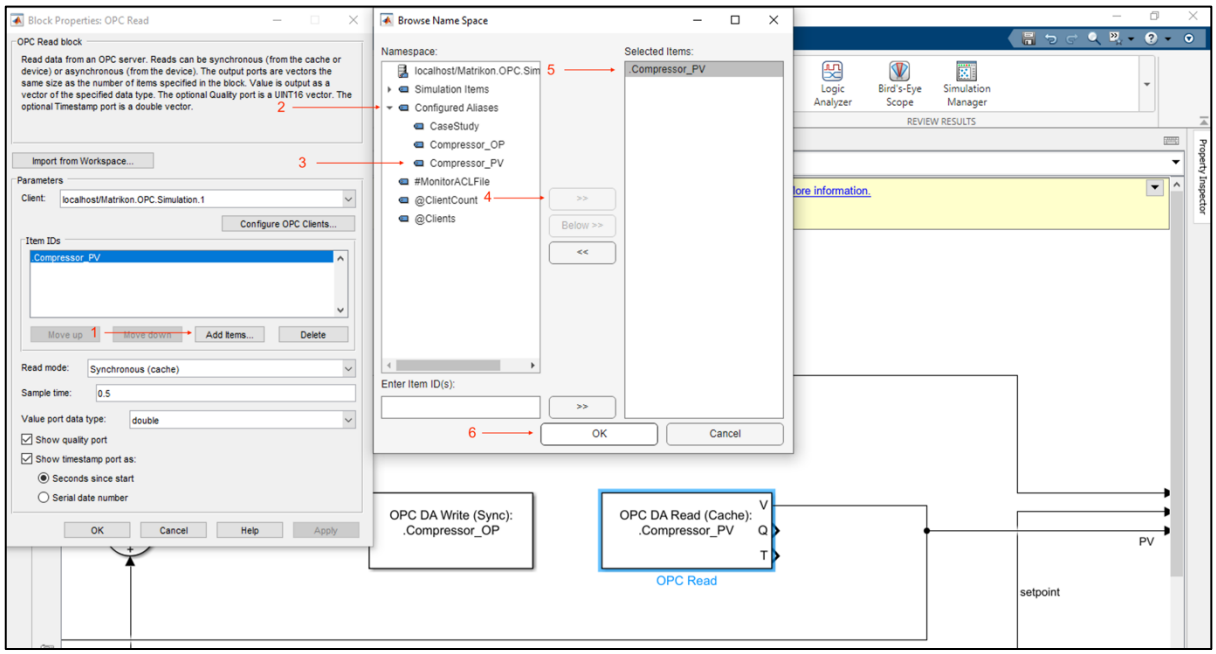


Figure 35 Simulink - OPC DA Read Configuration

14. As we will be using the controller which was developed in the Simulink to act as a PI controller in UniSim, the time constant in Simulink should be set to infinite, as shown in *(*Figure 36*)*.

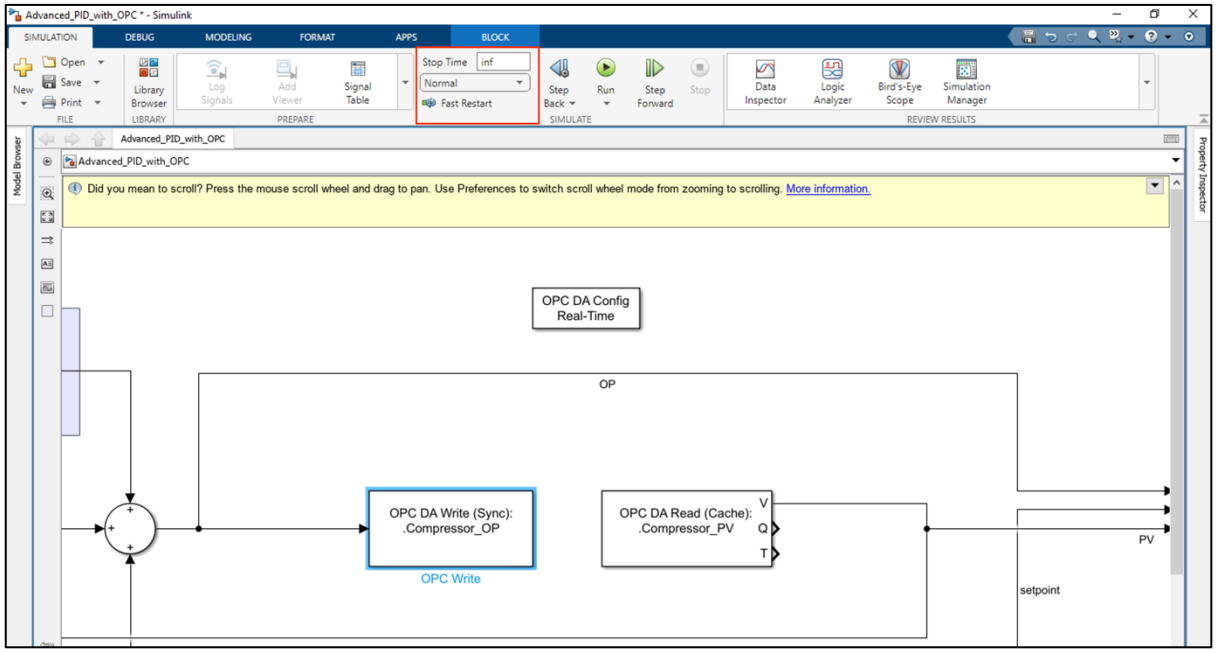


Figure 36 Simulink - Time Constant

5.8 Simulink PID with UniSim Heat Pump over OPC DA

In the final step, the performance of the controller developed in Simulink was tested directly with the real heat pump system (UniSim model) using OPC DA communication established. Figure 37shows the complete setup of the established communication between the controller developed in Simulink and the actual heat pump system in UniSim.

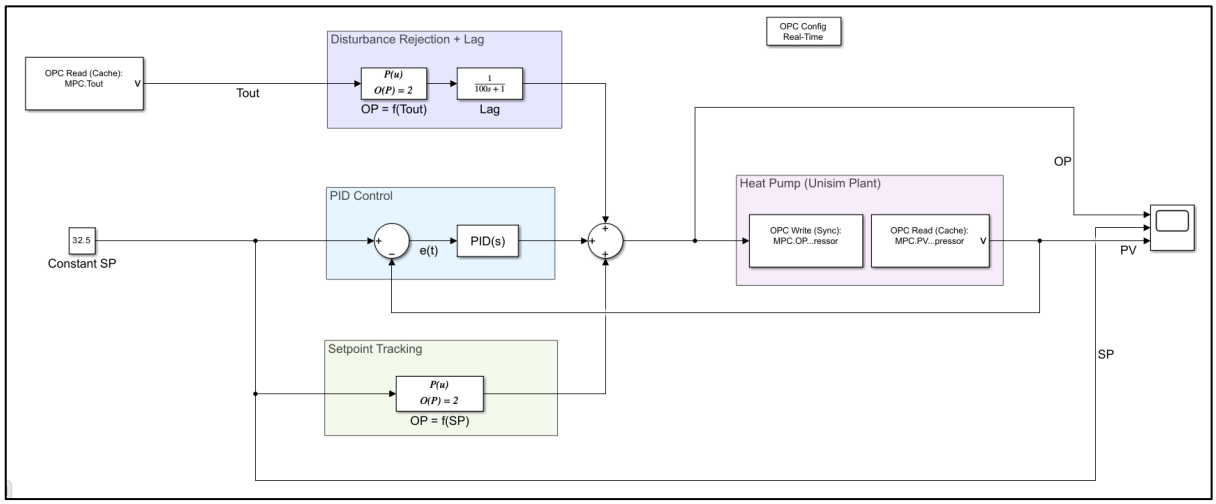


Figure 37. Communication between Simulink controller and UniSim model

6 Results and Discussions

6.1 Simulated behavior

Initially, the developed system model was tested with the proposed “PID with feedforward” controller. During the testing, an untuned PI controller with $K_P = 5$ and $K_I = 3$ was employed.

Firstly, control action was observed for the objective of driving the system temperature towards a constant desired setpoint of 32.5 ℃, in the presence of a constant atmospheric temperature $T_{out} = -10$ ℃ outside acting as the disturbance. From Figure 38, it can be seen that the proposed PI controller is able to maintain the setpoint and reaches the steady state in approximately $200\ s$.

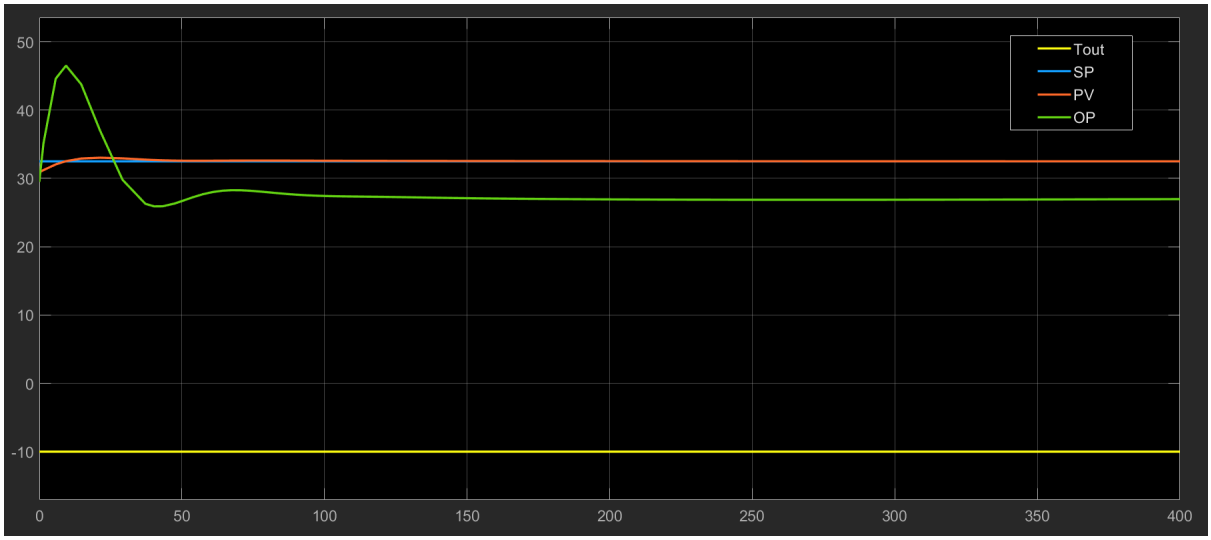


Figure 38 Simulated control action for constant setpoint and disturbance

Additionally, the control action was also observed for a changing setpoint and constant disturbance, and the observed behaviour was satisfactory. The controller was able to adjust the compressor speed quickly to react to the change, thus showing the advantage of having the setpoint tracking feedforward control.

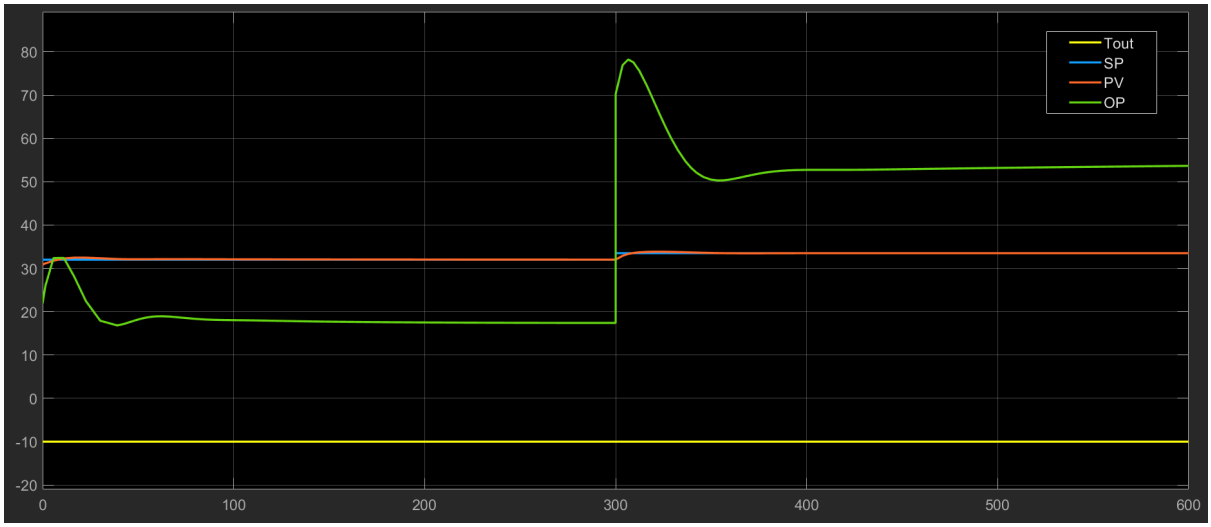


Figure 39 Simulated control action for changing setpoint and constant disturbance

In the final test, real-life scenario was attempted to be replicated by simulating a varying disturbance throughout the day. In geographic locations like Europe, it is very common to have atmospheric temperatures varying between day and night. This was simulated as a sinusoidal wave with its amplitude ranging from -10 ℃ to $+2$ ℃ over 36000 seconds.

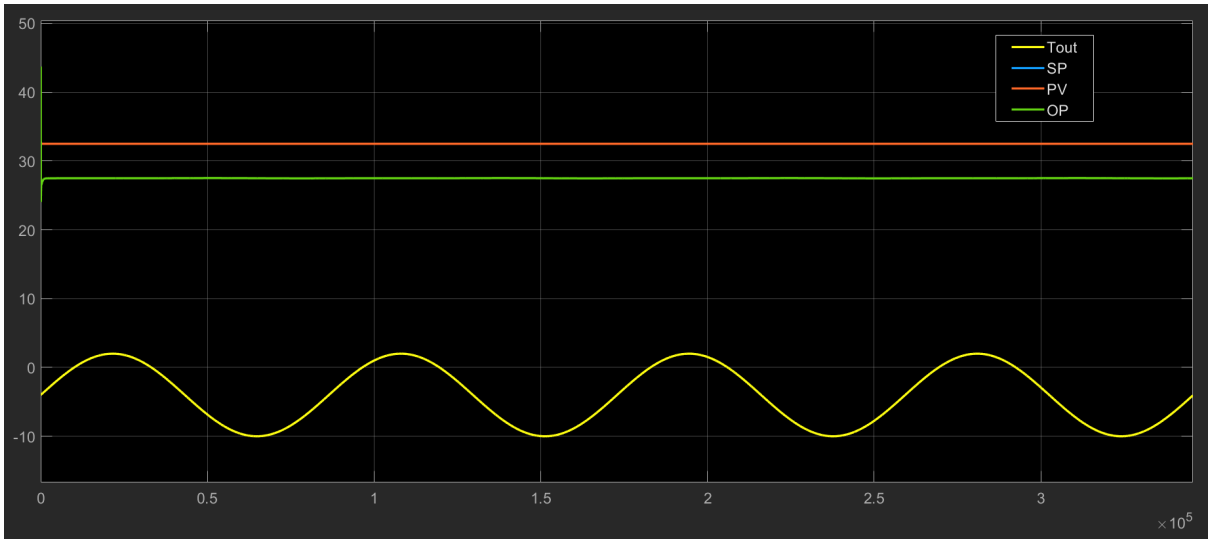


Figure 40 Simulated control action for constant setpoint and varying disturbance

The controller is able to maintain the temperature at a constant desired level by actively rejecting the effect of the disturbance variable.

Owing to these advantages, the proposed controller was then tested with the actual heat pump system, and its performance was analyzed.

6.2 Behavior of the proposed PID controller with the real system

6.2.1 Different gains for P and I

Given, that the actual system naturally behaves different than the simulated model, tests were performed to analyse the effect of different values of the P and I gains. The derivative term was not considered given the noisy measurement behaviour over OPC DA.

The tests were performed in the presence of a constant atmospheric temperature $T_{out} = -9$ ℃ (disturbance variable) and for varying setpoint temperatures. The figures given below illustrate these differences in the system behaviour. Figures given below illustrate these differences.

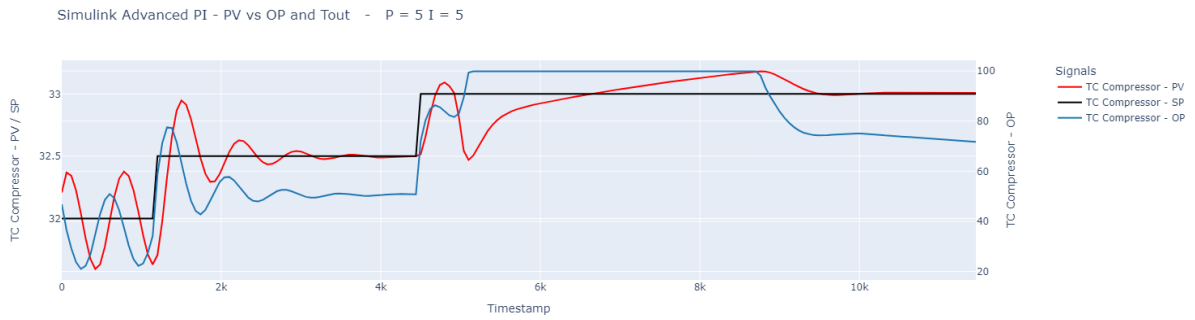


Figure 41. Simulink PI with P = 5 and I = 5

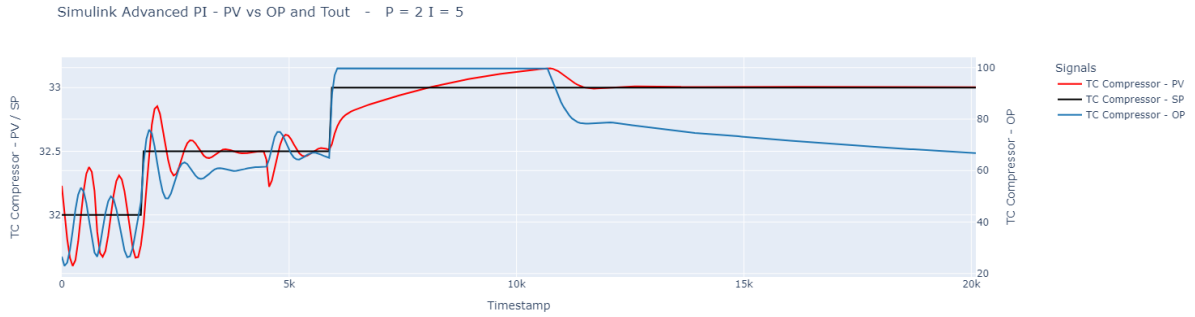


Figure 42. Simulink PI with P = 2 and I = 5

As expected, high values of K_I result in a highly oscillatory behavior, especially when the OP is running not too close to its upper limit. This is in alignment with the inherent nature of integral control, making the PI controller aggressive against the accumulating error.

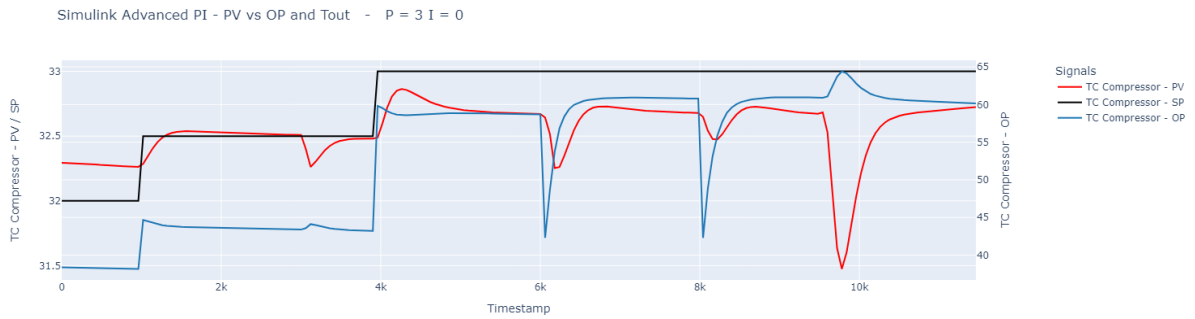


Figure 43. Simulink PI with P = 3 and I = 0

Employing only proportional control results in the presence of a steady-state error and the controller is unable to reach higher setpoints (along with the existence of inherent spikes in the plant behavior).

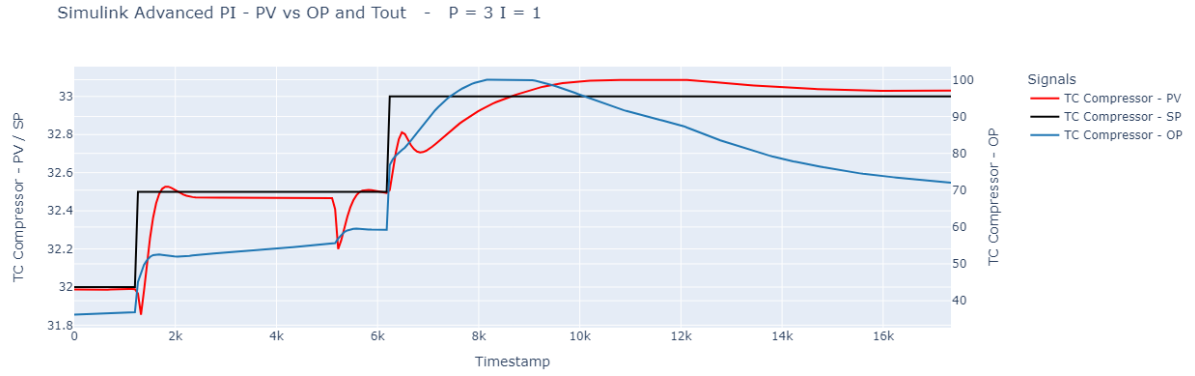


Figure 44. Simulink PI with P = 3 and I = 1

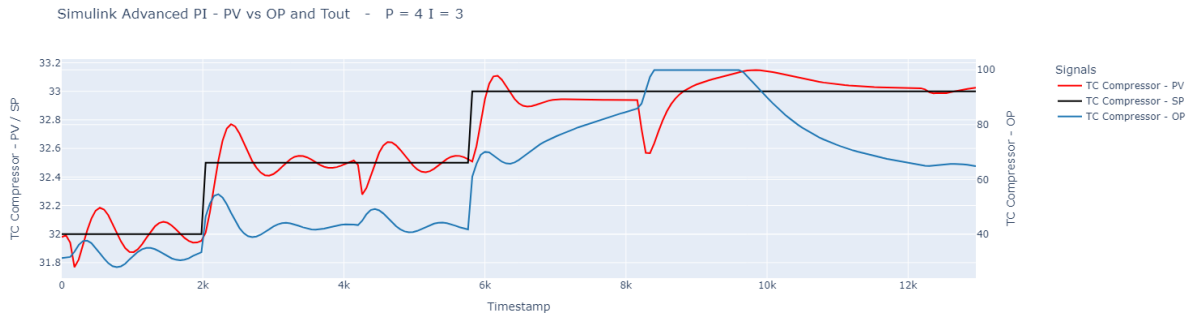


Figure 45. Simulink PI with P = 4 and I = 3

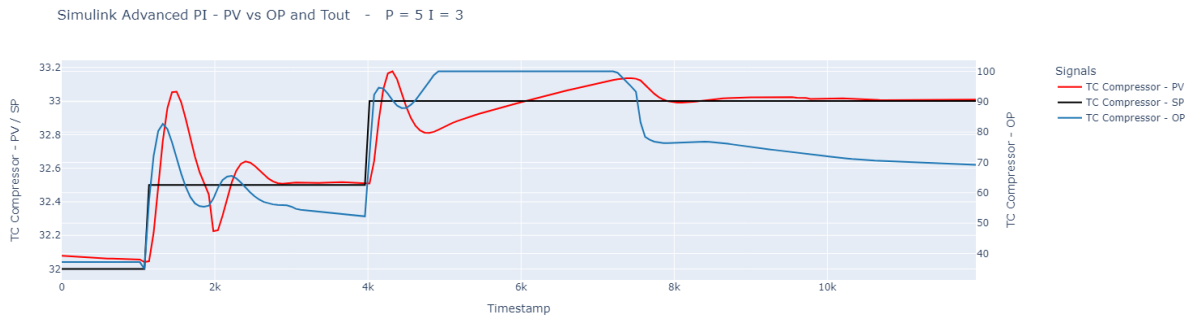


Figure 46. Simulink PI with P = 5 and I = 3

On further testing, the values $K_P = 5$ and $K_I = 3$ were selected since with these gains, the controller reacts fast enough against setpoint changes, while also having no so large overshoots.

6.2.2 Different Lag values

As stated earlier, in the feedforward control against the disturbance variable, a lag element is used in order to delay the controller reaction and not act too fast against the changes. Therefore, tests were performed to analyze the effect of the amount of lag. The conducted tests involved defining both a temperature set point and a constant ambient temperature. Subsequently, an abrupt change in the outside temperature was simulated to analyze the system's response to this variation, using different values for the lag element.

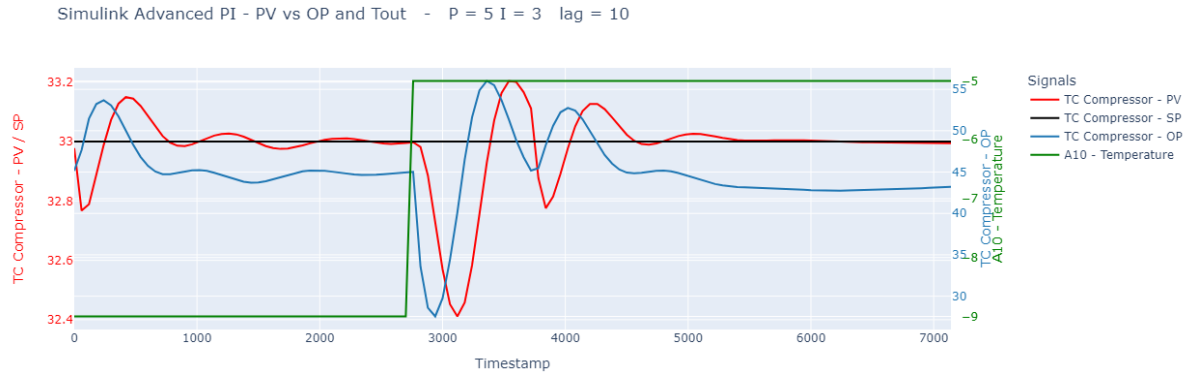


Figure 47. Simulink PI with P = 5 and I = 3 and Lag = 10

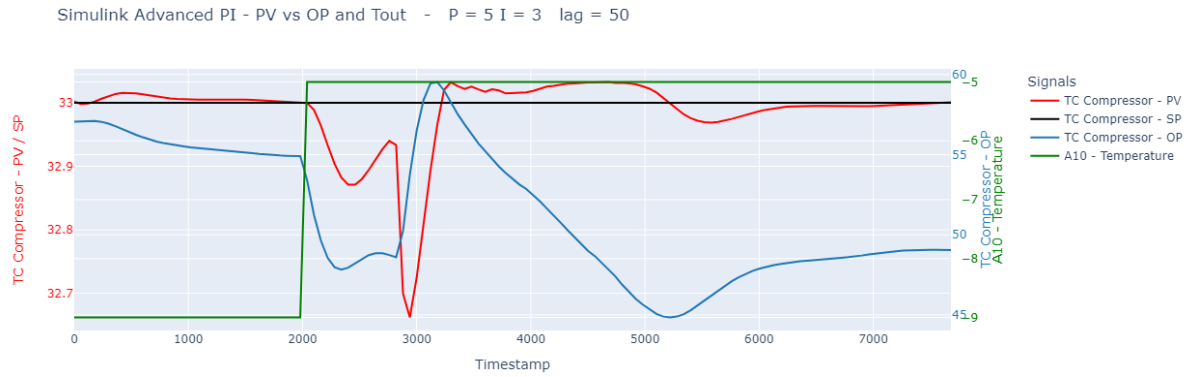


Figure 48. Simulink PI with P = 5 and I = 3 and Lag = 50

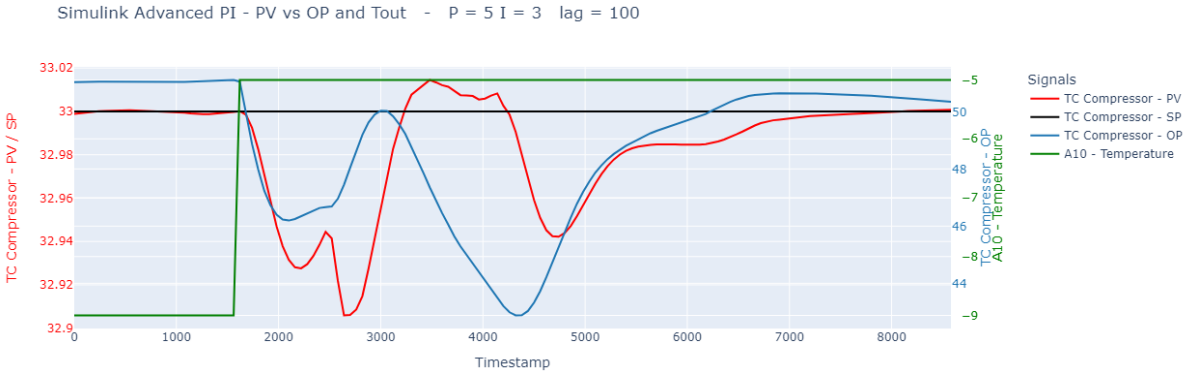


Figure 49. Simulink PI with P = 5 and I = 3 and Lag = 100

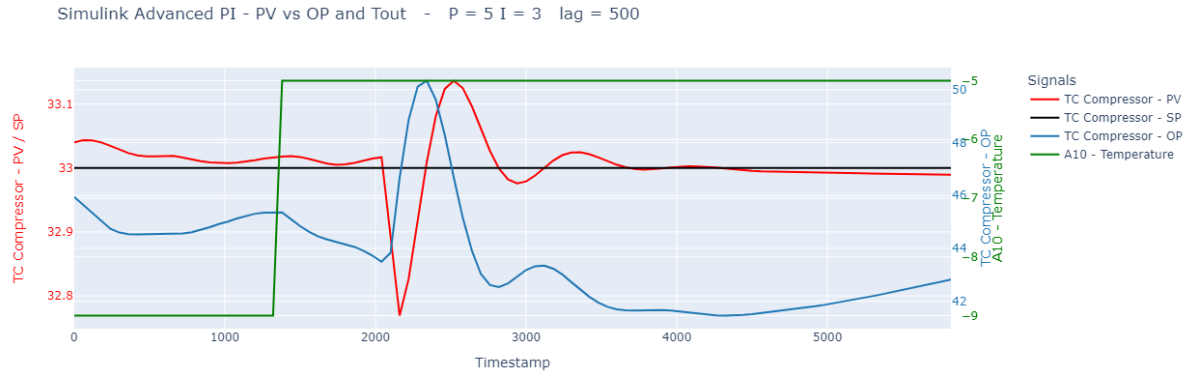


Figure 50. Simulink PI with P = 5 and I = 3 and Lag = 500

The different lag values tested were 10, 50, 100, and 500 seconds. Overall, the system performed satisfactorily for all these values, effectively reacting to the abrupt change in the outside temperature. However, a lag value of 100 was selected as the optimal choice because it resulted in the smallest variation from the defined set point.

6.2.3 Day/Night Simulation

Using the previously defined parameters (P = 5, I = 3, and lag = 100), another simulation was conducted. In this scenario, instead of an abrupt change in the outside temperature, it gradually altered over time to represent the fluctuations typically observed within a single day. Additionally, temperature setpoint adjustments were made to further analyze the model's behavior under these conditions. The image below represents a 24-hours simulation with a temperature variation from -11°C to -6.5°C, and changes in SP from 33°C to 33.5°C and then to 34°C.

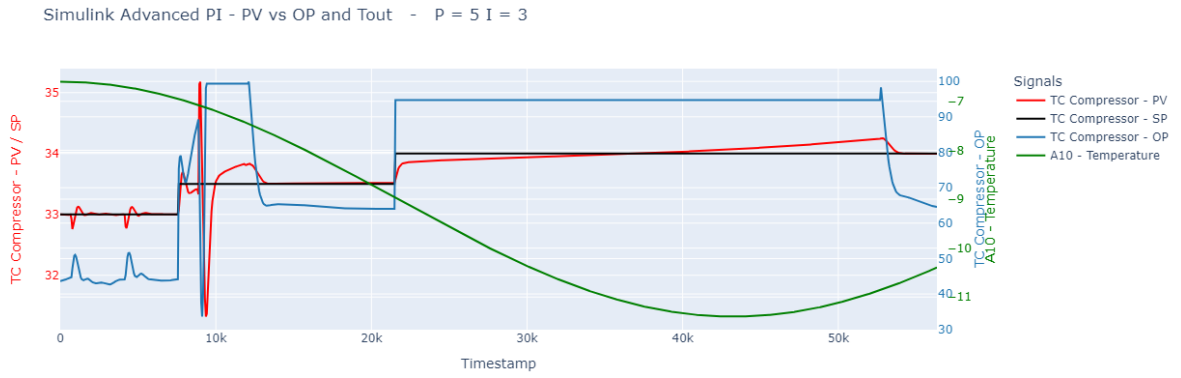


Figure 51. Day/Night Simulation

6.3 Simulink PID vs UniSim PID

Lastly, a final simulation was conducted to compare the behavior of both the Unisim and Simulink PI controllers. In this test, the outside temperature was altered similarly to previous simulations, and a temperature set point was defined. The performance of both controllers was then analyzed in terms of their ability to maintain the set temperature throughout the day.

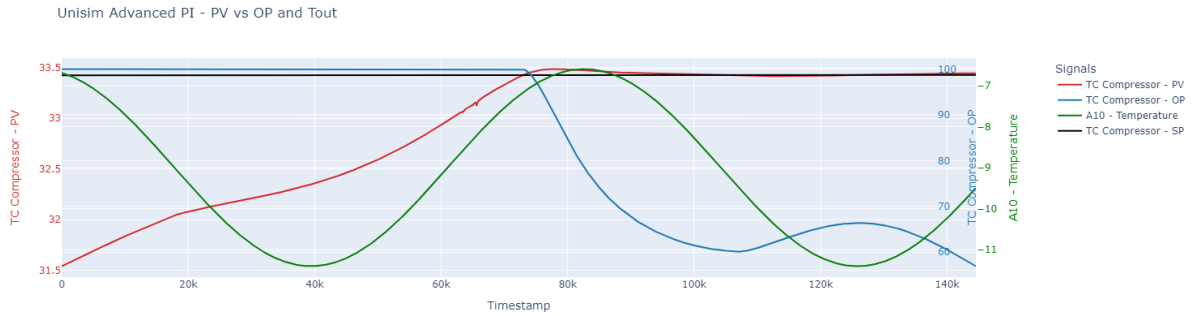


Figure 52. Unisim PI Day/Night Simulation

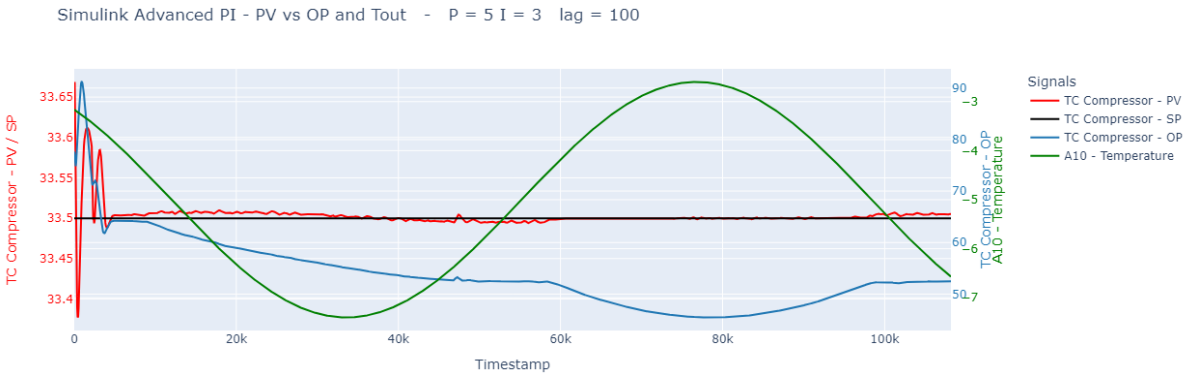


Figure 53. Simulink PI Day/Night Simulation

7 The Model Predictive Control Approach

Model predictive control (MPC) is an optimal control technique whereby computed control actions are applied to minimize an objective function of a constrained dynamic system over a finite, receding horizon.

At each time step, MPC models predict the change in the dependent variables of the modelled system that will be caused by changes in the independent variables. They use the current plant measurements, the current dynamic state of the process, the internal plant models, and the process variable targets and limits to calculate future changes in the dependent variables. These changes are calculated to hold the dependent variables close to target while obeying constraints on both independent and dependent variables. The controller then applies to the plant only the first computed control action, disregarding the following ones. In the next time-step, the process repeats. The diagram below concisely captures the operational method of a model predictive controller.

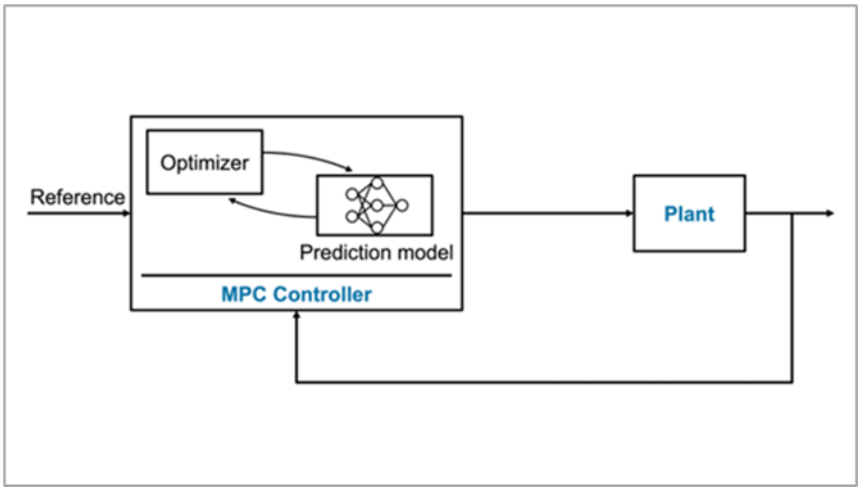


Figure 54 Schematic Representation of MPC

Model Predictive Control represents a significant advancement in control technology, offering a sophisticated approach to managing complex, multivariable systems. Its predictive nature, coupled with the ability to handle constraints and optimize performance, makes MPC a versatile and powerful tool across various industries. It has been successfully applied in a wide range of industries and applications, including but not limited to process control, energy systems, automotive industry, aerospace, robotics etc.

7.1 The Operational Principle of MPC

Model predictive Control is an iterative algorithm which tends to optimize the dynamic model of a plant within a finite horizon. At a given time t, the current state of the plant is determined, and an objective minimizing control strategy is computed for a relatively short time horizon in the future, $t+T$. The control method is applied only to the first step; after that, the plant state is sampled once more, and the computations are repeated from the new current state onward,

producing a new control and anticipated state route. Since the prediction horizon is constantly moving forward, MPC is also known as receding horizon control. Figure 2 below shows the mode of operation of MPC.

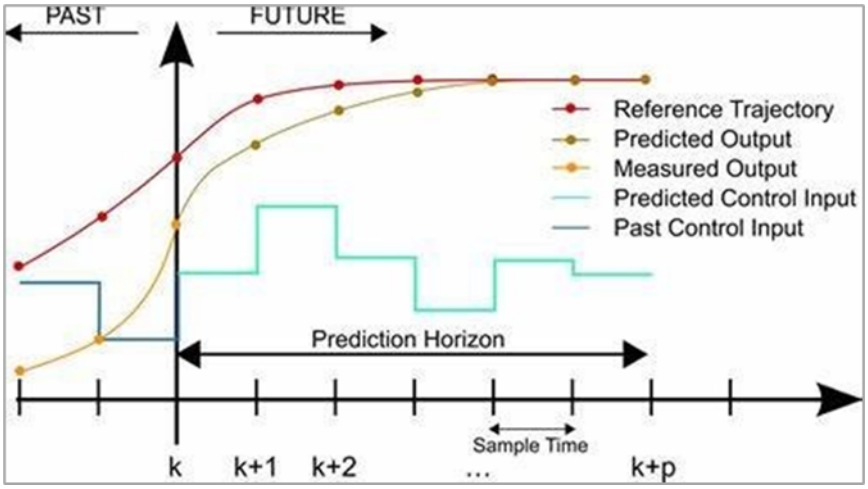


Figure 55 Operational Principle of MPC

7.2 MPC Terminologies

Some essential terms used in the context of MPC include:

15. **Prediction Horizon:** The prediction horizon is the finite future time over which the system's behavior is predicted using the model.
16. **Control Horizon:** The control horizon is the time span over which the control actions are optimized. It is often shorter than the prediction horizon.
17. **Cost Function (Objective Function):** The cost function is a mathematical expression that quantifies the performance of the control system. It typically includes terms for tracking error (difference between predicted and desired outputs) and control effort. The objective of MPC is to minimize this cost function subject to constraints.
18. **Constraints:** Constraints are limitations imposed on the control inputs, system states, and outputs. These can include physical limits, safety requirements, or operational boundaries. Constraints ensure that the control actions remain feasible and safe.
19. **Receding Horizon:** The receding horizon principle refers to the iterative nature of MPC, where the optimization problem is solved at each control interval using a moving time window.
20. **Model:** The model represents the mathematical description of the system’s dynamics. It can be linear or nonlinear and is used to predict future states of the system based on current states and control inputs.
21. **Optimization Solver:** The optimization solver is the algorithm or numerical method used to solve the MPC optimization problem. It determines the optimal sequence of control inputs that minimize the cost function while satisfying the constraints.

7.3 MPC Workflow in Simulink

In Simulink, modelling a model predictive controller (MPC) can be achieved in many ways. In this case study, we used the MPC Designer approach. Hence, this report contains only this method.

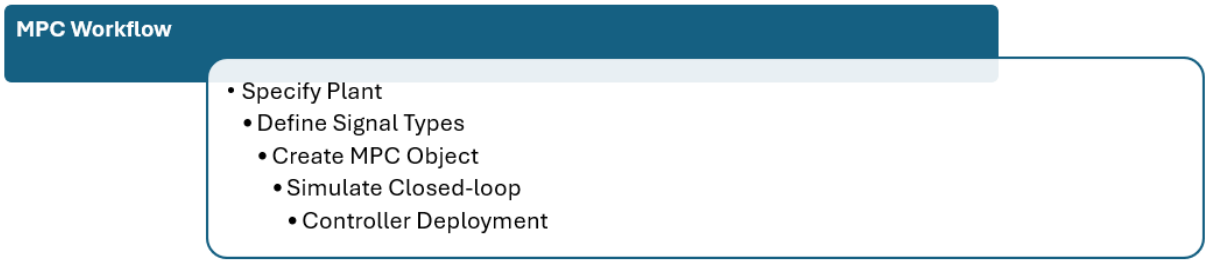


Figure 56 MPC Workflow

7.3.1 Specify Plant

A model predictive controller uses linear plant, disturbance and noise models to estimate the controller state and predict future plant outputs. The model of the Heat Pump which is our plant was modelled using the system identification toolbox (as described in PID part of this report above). To develop the MPC controller, we used the Model Linearizer App in Simulink. To use this linearization method, the input and output points of the plant must be specified. These points are referred to as the Linear Analysis Points. The input point is called the input perturbation (point A) while the output point is the open-loop output (point B) (figures are shown below). After specifying these points, the model linearizer can be applied to linearize the plant. The linearized model is obtained as *linsys1*.

Model_Linearizer

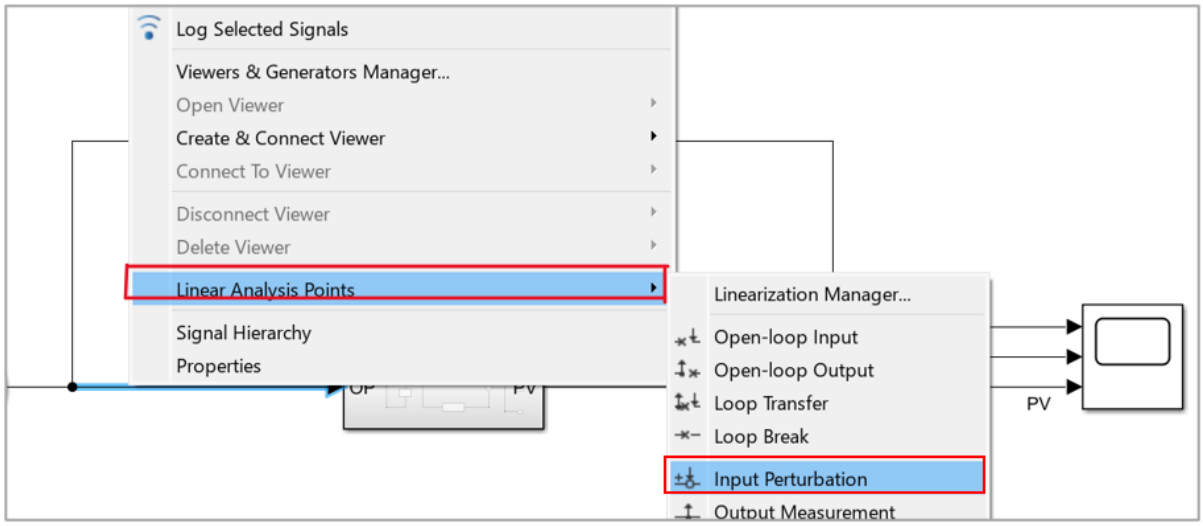


Figure 57 Input Perturbation

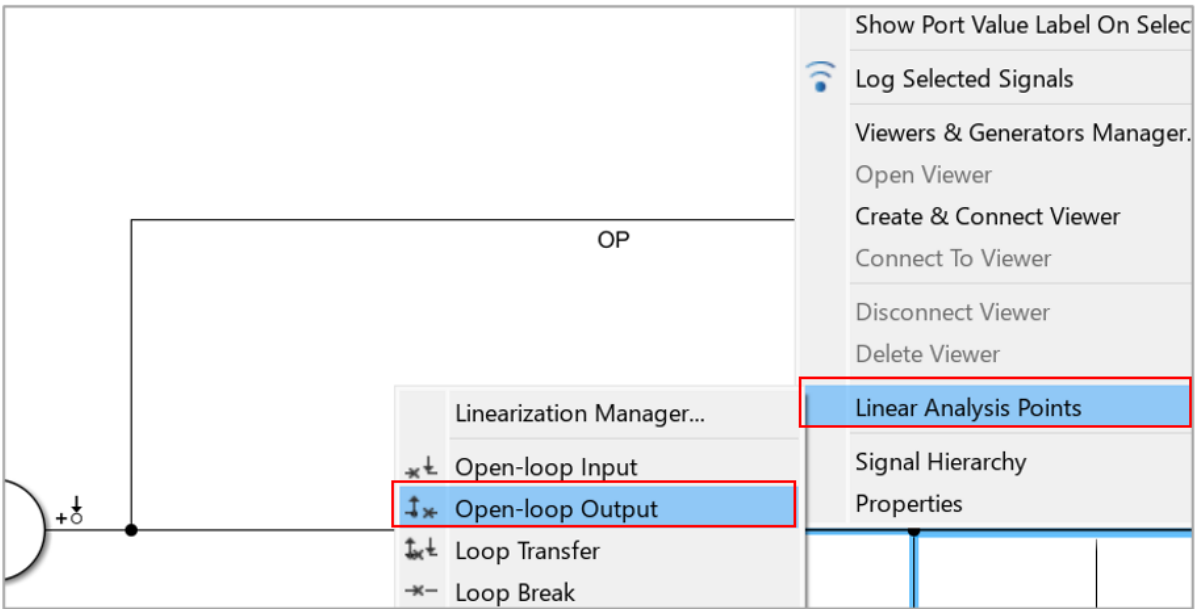


Figure 58 Input Perturbation

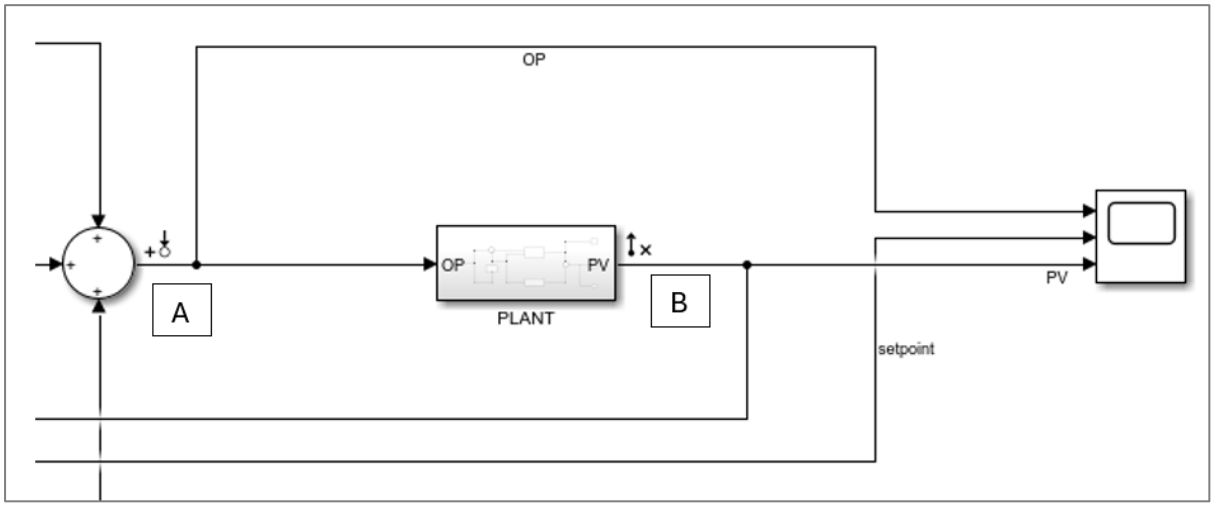


Figure 59 Linearized between Points A and B

This App linearizes a nonlinear plant at a given operating point and specifies it as a LTI object, such as steady state (ss), transfer function (tf) or zero-pole-gain (zpk) models. The plant model was linearized using the App and the equivalent LTI model was obtained in *ss* and tf as shown below.

7.3.2 State Space

The linearized model in state space was obtained in the form:

$$linsys1.A = [-0.1,0,0;\ (7.28\ x\ 10^{-7}),0,0;\ (-2.267\ x\ 10^{-4}),0,-0.033]$$
$$linsys1.B = [1;\ (1.0\ x\ 10^{-6});\ (6.48\ x\ 10^{-4})]$$
$$linsys1.C = [0.00693,1,1]$$
$$linsys1.D = 0$$

7.3.3 Transfer Function

The equivalent transfer function of the linearized model is given below:

$$H(s) = \frac{(7.579\,x\,10^{-3})s^2 - (1.38\,x\,10^{-4})s + 2.76\,x\,10^{-9}}{s^3 + 0.1033s^2 + (3.33\,x\,10^{-4})s}$$

7.3.4 Response Plots

Response plots of the output to the internal plant model can also be obtained using step-input, Bode plot, Nyquist plot etc. From the step input plot below, the output signal tracks the reference input. This depicts good performance by the designed controller. The Bode plot is also shown below.

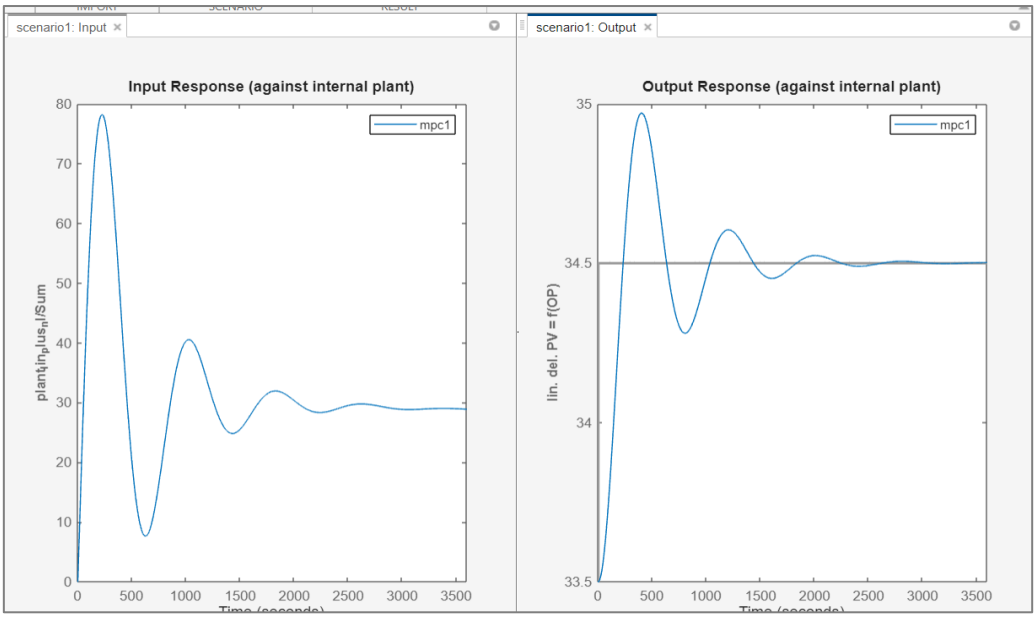


Figure 60 Step Response Against the Linearized (internal) Plant

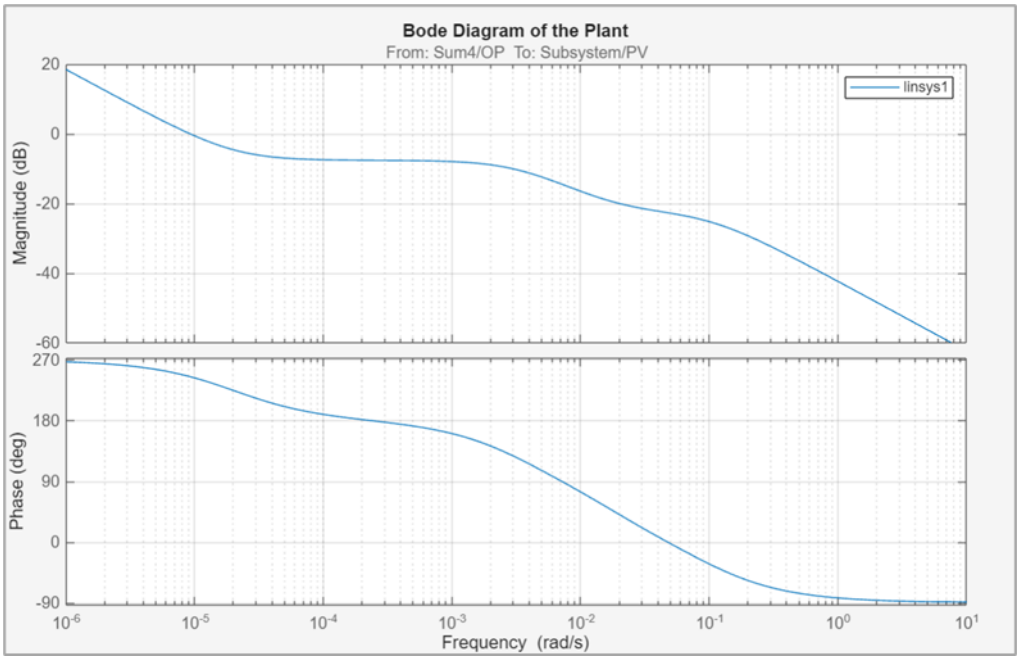


Figure 61 Bode Plot of the Linearized (internal) Plant

7.3.5 Define Signal Types

The next stage is to import the linearized plant model into the *mpc Designer* APP. This is the starting point of designing mpc controller. In MPC design, plant signals are usually classified into different input and output types depending on whether each plant output is measured or unmeasured, and whether each plant input is a manipulated variable (that is, a control input) or a measured or unmeasured disturbance. In our experiment, the OP (compressor speed) was defined as a manipulated variable while the PV (room temperature) was defined as a measured output (see diagrams below). Since we planned to incorporate the ambient temperature/disturbance (Tout) at a later stage, we did not explicitly define it here. Hence, we have 1 input, 1 output and 3 states as obtained after the linearization.

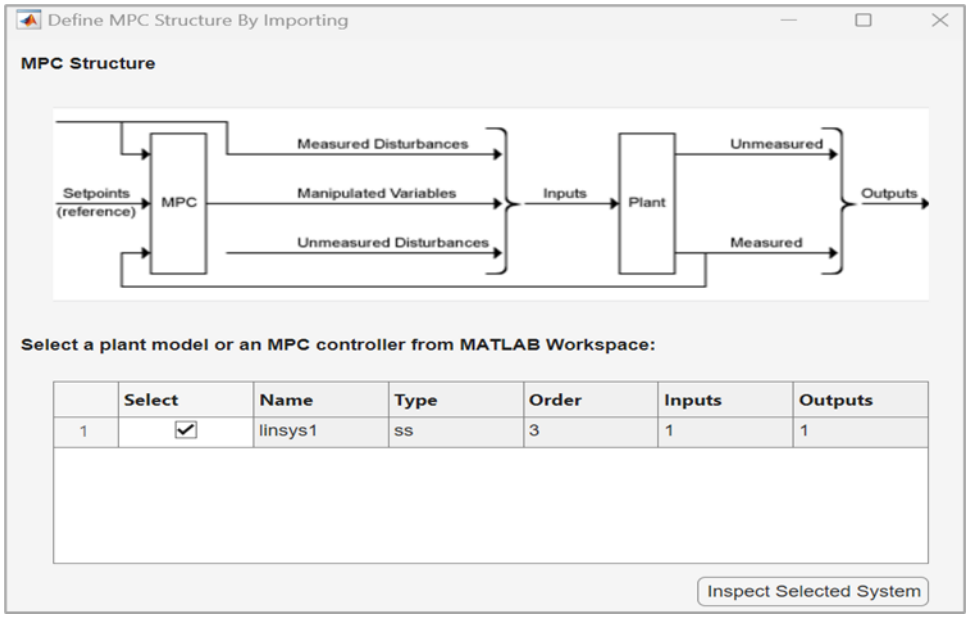


Figure 62 Various Signal Types for Defining MPC

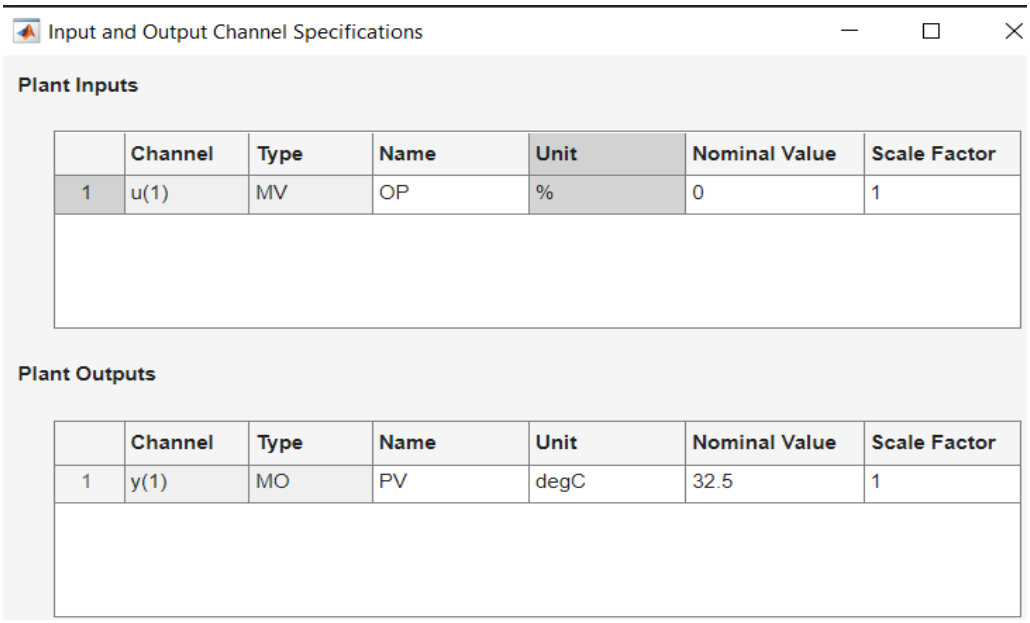


Figure 63 Various Signal Types for Defining MPC

7.3.6 Create MPC Object

The next stage is to create an MPC object *(mpc1)* in the MATLAB workspace or in the MPC designer and specify, in the object, controller parameters such as the sample time, prediction and control horizons, nominal values, scale factors, cost function weights, constraints, and disturbance models. Optimum values used are shown in the figure below.

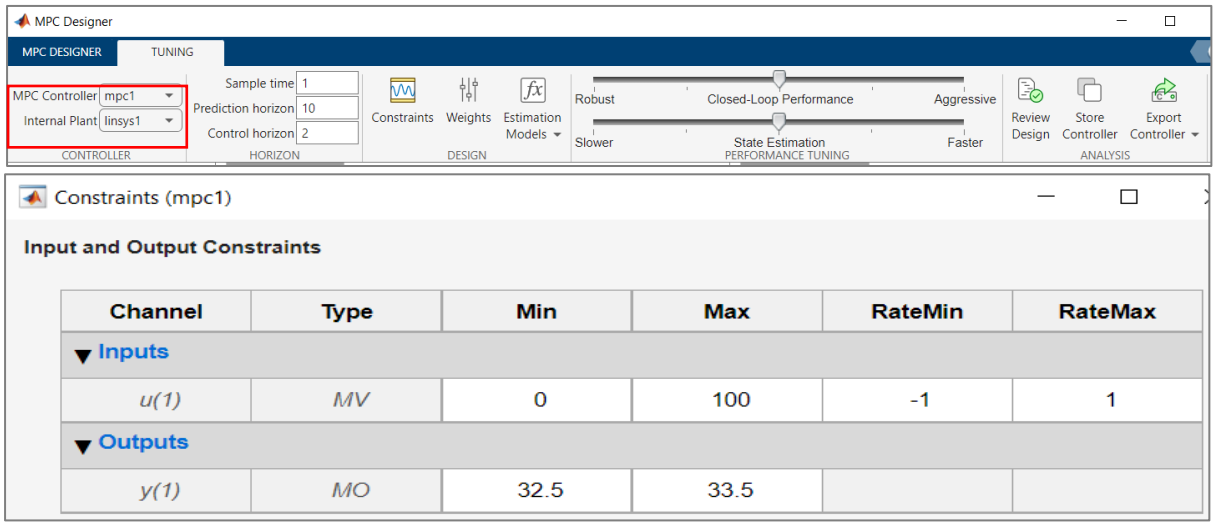


Figure 64 MPC Object (mpc1) and Constraints

7.3.7 Simulate Closed loop

Using the *mpc* object created, the next step is to typically evaluate the performance of the controller by simulating it in closed loop with the plant. We used the MPC controller block in Simulink (which takes the *mpc* object as a parameter) in closed loop with the plant to evaluate the performance of the controller, as shown in the diagram below. It is observed that while the compressor speed rises and remains within the defined MV range of 0-100%, the MPC controller makes the room temperature PV to track the setpoint and remain within the defined boundary. Figure 65 and Figure 66 clearly describe this.

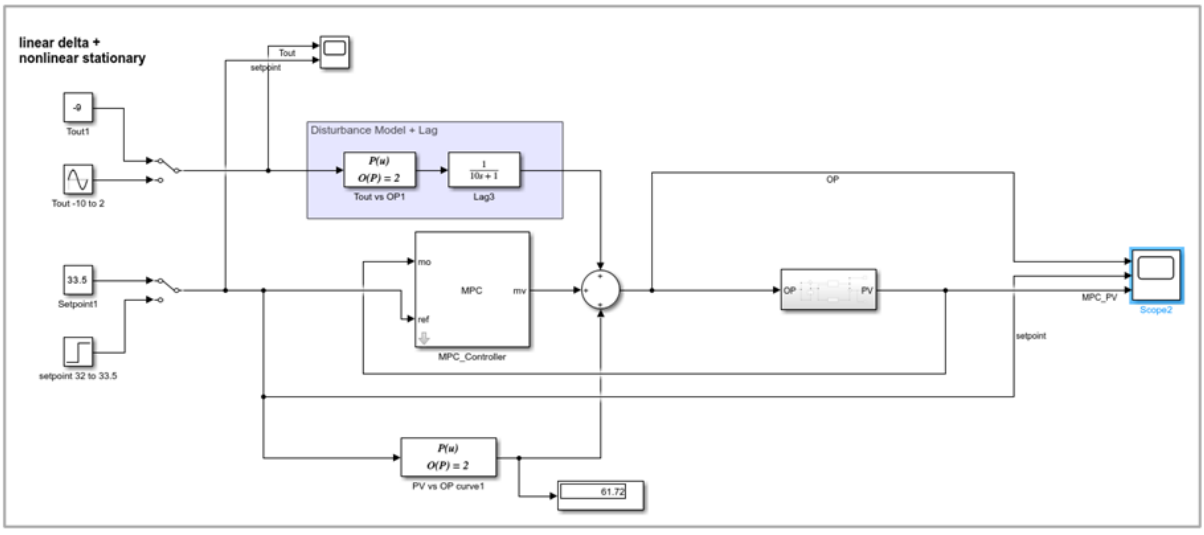


Figure 65 The MPC in Closed loop with the plant

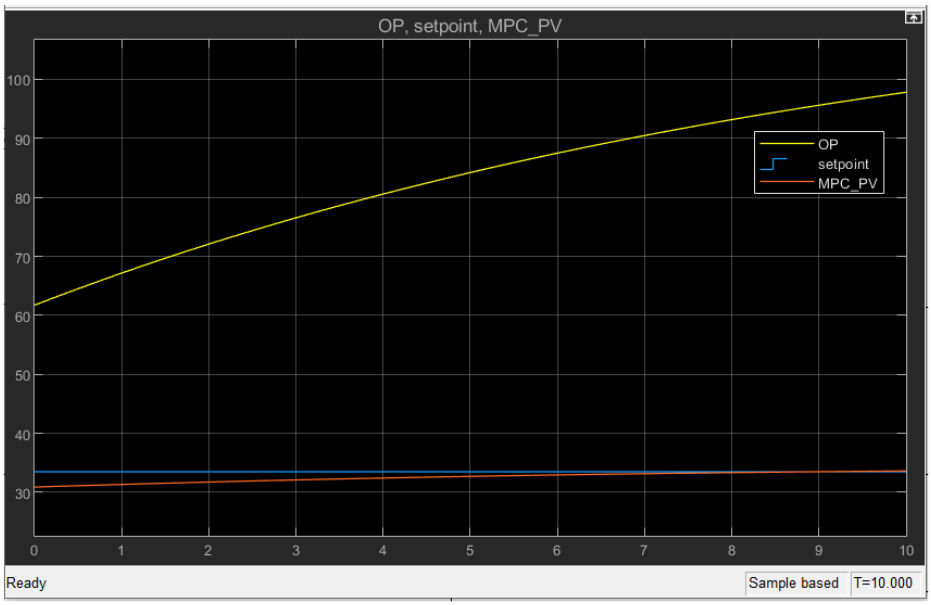


Figure 66 Closed loop simulation Response

7.3.8 Controller Deployment

The last stage in the experiment was to use the designed MPC controller in Simulink to control the plant in UniSim. To do this, we established a communication link between both entities using the OPC server/client communication tool (Matrikon software) as shown in *(*Figure 67*)* and (Figure 68*)*. The first test was carried out at the setpoint of 32°C and constant Tout of –9°C. The PV value obtained in the Matrikon and Unisim software is 32.9°C. This value was obtained after a run time of about 1hr.

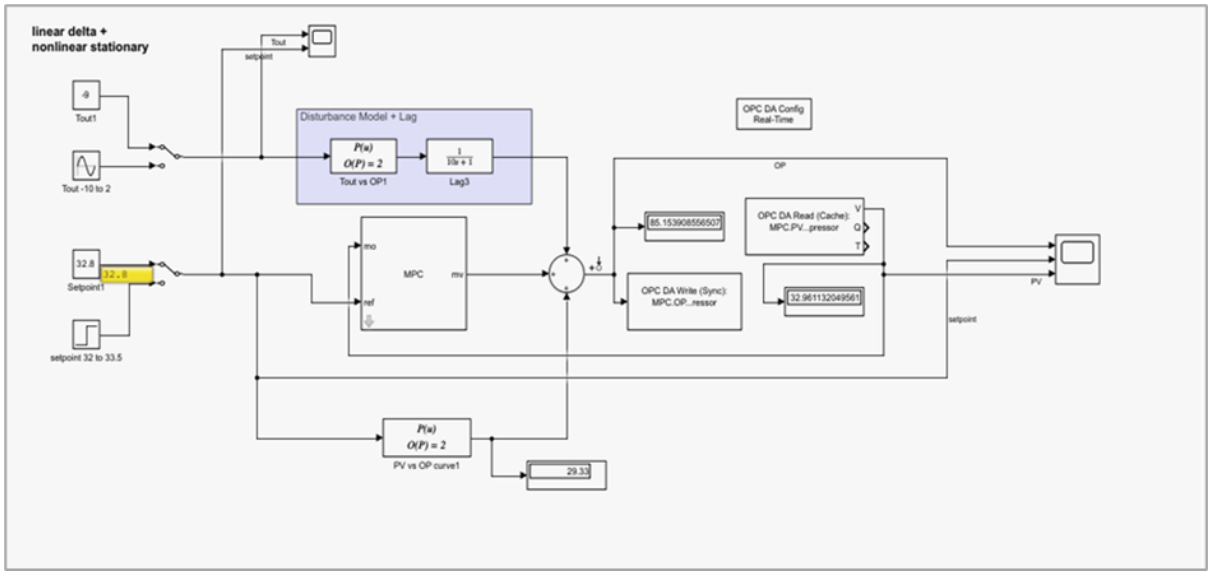


Figure 67 MPC deployment in Simulink

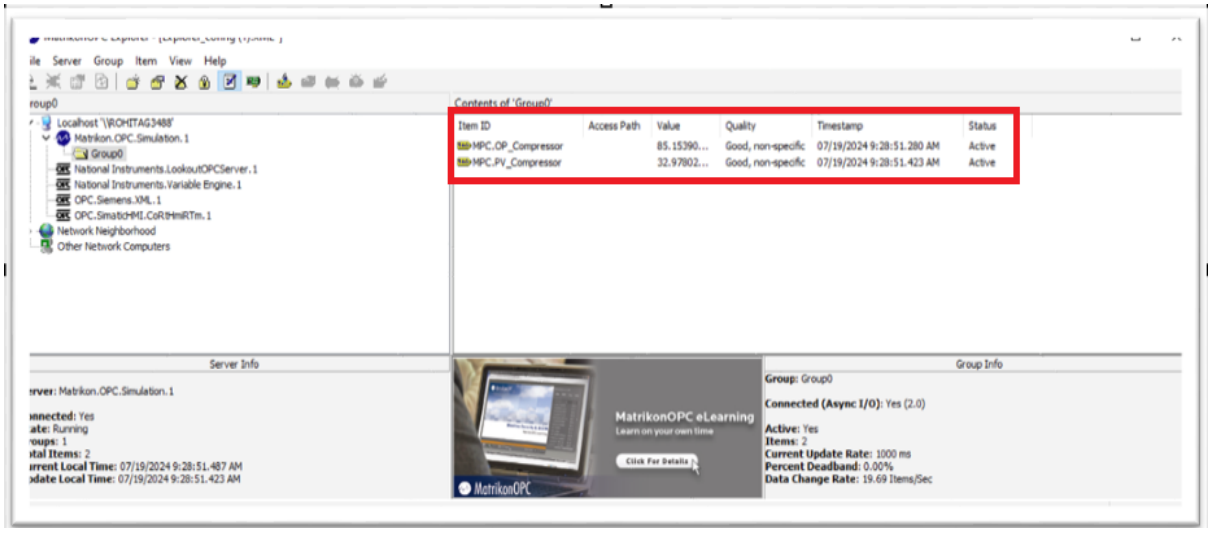


Figure 68 OPC DA Communication Channel between Simulink and Unisim Models

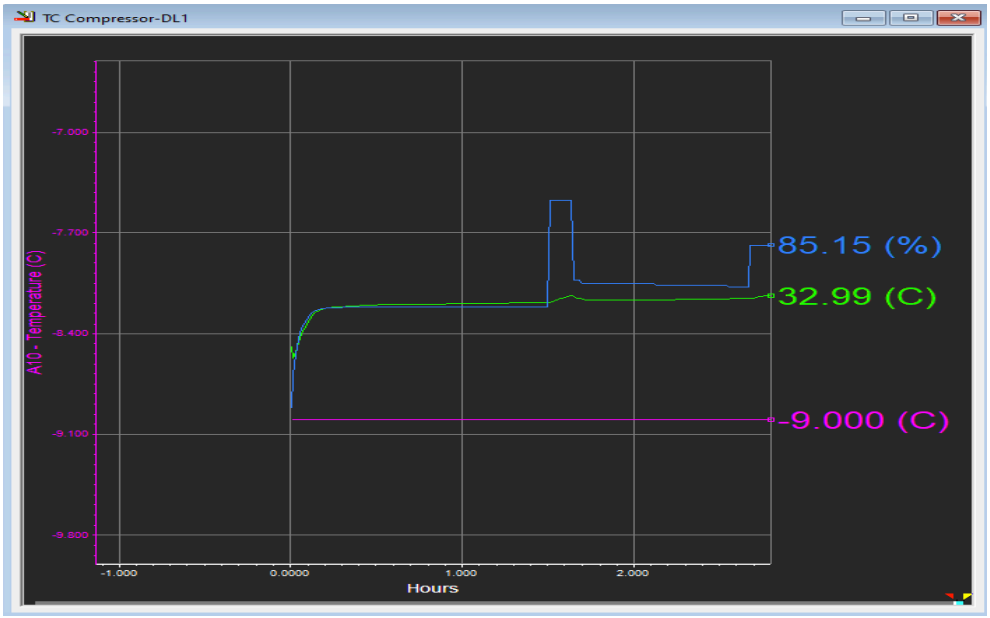


Figure 69 Unisim OP and PV values at Constant Tout (-9°C)

It was also observed that as time continued to increase, the PV also continued to rise. For instance, it can be seen from the figure below that at about 52hrs run time, the PV had increased to 36.7°C. Hence, the MPC did not achieve the desired control results.

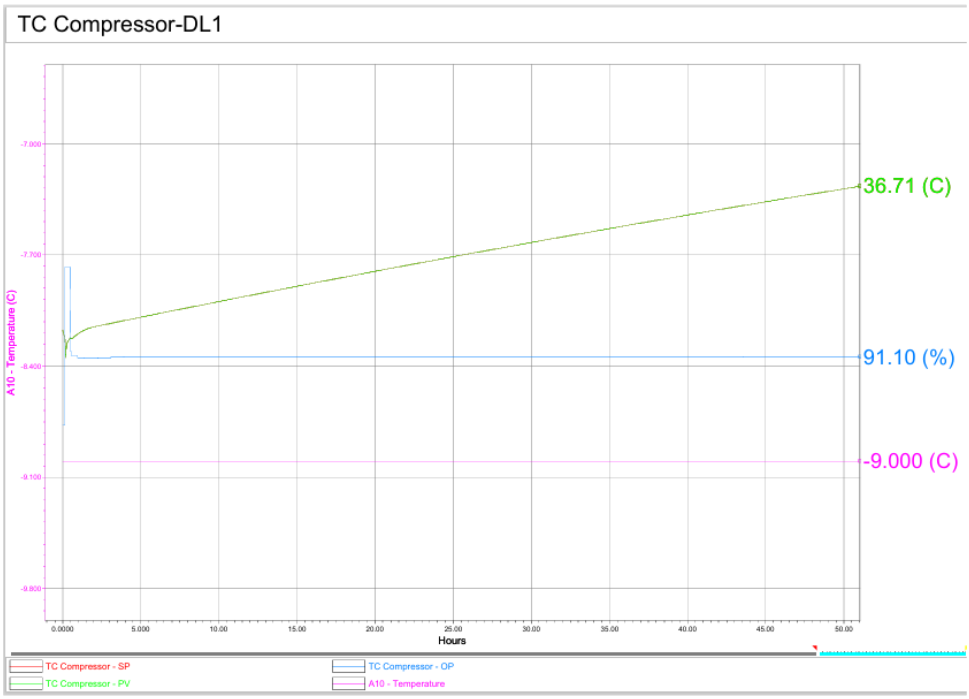


Figure 70 OP and PV values at 52hrs run time

8 Conclusion

This project involved the development of both advanced PI and MPC control algorithms with which to control the heating of a room. The impact of the outside temperature (Tout) was also considered, and the robustness of the developed algorithms were evaluated. While the advanced PID performed well by rejecting the disturbance (when modeled both as a constant and as a day-night effect) and maintaining the preset temperature, the MPC did not yield expected results as it was not able to maintain the room temperature at the determined set value. We acknowledge that we were limited in understanding the intricacies of MPC's operations. Further work on it with expert knowledge will enhance its performance.

It was a very rigorous experiment as we were confronted with many challenges including having inadequate knowledge of how the whole system should work and hence the volume of time spent. However, the results and the progress made in developing both algorithms from scratch have given us invaluable insights into such systems and will be helpful to us in our future engagements.

9 Bibliography

[1] T. A. Deetjen, L. Walsh and P. Vaishnav, "US residential heat pumps: the private economic potential and its emissions, health, and grid impacts," *Environmental Research Letters,* vol. 16, p. 084024, 2021.

[2] M. U. Kajgaard, J. Mogensen, A. Wittendorff, A. T. Veress and B. Biegel, "Model predictive control of domestic heat pump," in *2013 American control conference*, 2013.

[3] T. Q. Péan, J. Salom and R. Costa-Castelló, "Review of control strategies for improving the energy flexibility provided by heat pump systems in buildings," *Journal of Process Control,* vol. 74, p. 35–49, 2019.

[4] A. E. Gürel and I. Ceylan, "Thermodynamic analysis of PID temperature controlled heat pump system," *Case Studies in Thermal Engineering,* vol. 2, p. 42–49, 2014.

[5] M. Akmal and B. Fox, "Modelling and simulation of underfloor heating system supplied from heat pump," in *2016 UKSim-AMSS 18th International Conference on Computer Modelling and Simulation (UKSim)*, 2016.

[6] M. Inc., 31 10 2002. [Online]. Available: https://www.matrikonopc.com/training/opc-demo-tutorial.pdf.

[7] M. A. Salam and M. M. Islam, *Modelling and Control System design to control Water temperature in Heat Pump,* 2013.

[8] P. Byrne, J. Miriel and Y. Lénat, "Modelling and simulation of a heat pump for simultaneous heating and cooling," in *Building simulation*, 2012.

[9] M. W. Ahmad, "Advanced control strategies for optimal operation of a combined solar and heat pump system," 2013.