CRANFIELD UNIVERSITY

BEATRICE FANIYI

IOT ENABLED GREENHOUSE AUTOMATIC CONTROL SYSTEM FOR ENERGY EFFICIENCY OPTIMIZATION

SCHOOL OF WATER, ENERGY AND ENVIRONMENT
PhD IN ENERGY AND POWER

PhD
Academic Year: 2019 - 2022

Supervisor: Dr Jerry Luo
Associate Supervisor: Prof. Patrick Luk
Feb 2022

CRANFIELD UNIVERSITY

SCHOOL OF WATER, ENERGY AND ENVIRONMENT
PhD in Energy and Power

PhD

Academic Year 2019 - 2022

BEATRICE FANIYI

IOT ENABLED GREENHOUSE AUTOMATIC CONTROL SYSTEM FOR ENERGY EFFICIENCY OPTIMIZATION

Supervisor: DR JERRY LUO
Associate Supervisor: PROFESSOR PATRICK LUK

This thesis is submitted in partial fulfilment of the requirements for the degree of PhD

ABSTRACT

Agricultural greenhouses provide optimal conditions for plant growth, but they consume an excessive amount of energy, making energy the second-largest expense after labour costs. Most of the energy is used for heating, which is a major contributor to the high energy demand of the system. Precise and timely control technology can help reduce energy costs and increase profitability. The integration of IoT into greenhouses is a new development in smart agriculture that has the potential to optimise energy use.

Various methods exist for optimising energy use in greenhouses, including the use of phase change materials, efficient greenhouse construction designs, and control systems. However, smart automatic control systems are an efficient method that has not been explored enough. Understanding the control algorithm and its proper implementation for use in the greenhouse control system is critical for energy optimisation.

This thesis makes three main contributions to greenhouse temperature control. First, a dynamic, physics-based model of greenhouse temperature was optimised to be adaptable for greenhouses equipped with IoT hardware. Second, two control algorithms were implemented in simulation to regulate the system to the grower's desired temperature, while four other control algorithms were implemented to evaluate their energy minimization capability. Results showed that the MPC controller was the best controller in terms of energy savings. Nevertheless, for small to medium greenhouse operators who may have limited resources, relatively simple on-off control algorithm is cost-effective. Finally, the study demonstrates that an IoT-based control system can optimise the energy use in the greenhouse.

The use of IoT technology has the capacity to overcome the greenhouse energy management problem with a distribution control system aided by cloud computing. This study demonstrates the potential of IoT-based control systems to save energy and improve greenhouse efficiency by reducing delays and increasing control effectiveness.

Keywords:

IoT (Internet of Things), greenhouse, optimization, energy efficiency, Physics-based, FOPDT model, temperature control, On-off control, PI control, MPC control, control system and experimental study.

ACKNOWLEDGEMENTS

I begin by giving thanks to the Almighty God, the giver of all wisdom and strength, for seeing me through the writing of this thesis. His guidance, discipline, and blessings have been invaluable, and I am forever grateful.

To my first supervisor, Dr. Jerry Luo, I extend my deepest appreciation for your unwavering support and guidance throughout this process. Your time, continuous support, invaluable advice, and patience were instrumental in shaping this thesis, and I am grateful for the opportunity to have worked with you. I would also like to thank my second supervisor, Prof. Patrick Luk, for his contributions and valuable feedback. Although you are not always available for meetings due to your other engagements, your suggestions are much appreciated.

To my husband, Oluwasesan Anu, thank you for your pleasant support and encouragement throughout this journey. Your love and assistance helped me stay motivated and focused. To my children, Lydia, Gbenga, and Deborah, your patience and understanding during the long hours of study are greatly appreciated. You are my inspiration and motivation.

I would like to extend my gratitude to my colleague Fergus, Zaharaddeen, and others who provided invaluable support in the laboratory during my practical work. Your assistance and expertise were critical to the success of my research, and I am grateful for your contributions.

To my dear mother, thank you for your constant prayers and support. Your love and encouragement have been a source of strength throughout this journey. I acknowledge the memory of my late father. Thank you, sir.

I am also grateful to my spiritual fathers, Pastor Olubi, Mummy, and the GMJ family; you are deeply appreciated, and I love you all. Your prayers and meetings are a source of strength that kept me going during difficult times.

Finally, I would like to express my love and appreciation to my siblings. Your love means a lot to me, and I am grateful for the bond we share. Thank you all for your contributions to the successful completion of this thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF EQUATIONS

LIST OF ABBREVIATIONS

CHAPTER 1 General Introduction

1.1 Greenhouse energy use and energy-saving techniques

With the world's population projected to reach 9.1 billion by 2050 (Sreekantha, 2016), the demand for sustainable and energy-efficient farming practises continues to increase. Greenhouse systems offer a solution to the challenge of producing high-quality crops to meet the needs of a growing population. As an important part of food production systems around the world, greenhouse agriculture allows growers to control the environmental and biological factors that affect crop growth, consequently ensuring optimal growing conditions year-round (Y. Li et al., 2018). Given the benefits of greenhouse agriculture, there is a growing interest in developing smart greenhouse technologies.

Although agricultural greenhouses offer a perfect environment for plant growth and food production and enable plants to be cultivated in seasons that would have otherwise prohibited growth (Guo et al., 2021), greenhouse energy consumption has increased sixfold over the last 20 years (McNulty, 2017). The cost of heating and cooling greenhouses accounts for approximately 70–85% of total operating costs, excluding labour. Thus, reducing the energy consumption of greenhouse systems is crucial for managing agricultural greenhouses due to rising energy prices, a shortage of energy reserves, and environmental problems such as ozone layer depletion, global warming, and climate change. The use and constant improvement of automated agriculture technologies can play a crucial role in meeting the anticipated global food demand and creating an opportunity to cultivate cash crops for economic benefits.

As discussed above, although greenhouses are one of the most lucrative sectors in the agricultural industry, high cultivation output requires a significant capital investment in fertilizers, labour, and energy input for lighting and heating. With the continuous increase in energy demand, particularly for fossil fuels, reducing the required energy need is crucial to decreasing the total annual operating cost (Taki, Rohani, and Rahmati-Joneidabad, 2018).

Additionally, it is expected that the impacts of climate change on livelihoods and agricultural production will accumulate over time. The goal to reduce emissions by about 70% by 2050 to manage climate change (McNulty, 2017) can only be achieved with the help of the agricultural sector, which is responsible for about 20% of all emissions right now. As a result, any gains in energy efficiency can significantly contribute to the desired economic performance of greenhouse crop production and align with the goal of reducing emissions efficiently into the environment.

The cost involved in heating (primarily) and cooling the greenhouses accounts for about 70–85% of the overall operating costs, excluding labour costs (Ahamed et al., 2019). Approximately 65–85% of the total energy consumed in greenhouses is used for heating, while the remaining portion is used for transportation and electricity (Ahamed et al., 2019). In warmer regions, the cost of heating (specifically) and cooling a greenhouse can amount to around 50% of the total operating cost. In northern latitudes, the cost can increase to 70-85% of the total operating cost. Therefore, it is evident that heating and cooling costs make up a significant portion of the greenhouse's operating budget. Implementing measures to reduce the cost of these environmental factors will increase profits (Rorabaugh, Jensen, and Giacomelli, 2002).

An identified approach to improving greenhouse energy efficiency is through the exploration of a control strategy for optimal indoor regulation. The greenhouse control system consists of two parts: the hardware (sensors, controllers, and actuators) and the software (algorithm). According to (S. Zhang et al., 2020a), the precision of the control system hardware indirectly impacts energy consumption. Thus, an effective control system that is smart in coordinating the sensors and actuators of the control system hardware by improving the performance of the control algorithm to maintain the microclimate has the potential to significantly reduce energy costs. The greenhouse environment is extremely dynamic and complex, and automating it is becoming increasingly essential for the profitability and success of the system due to the pressures of cost, labour availability, market demands, and energy costs.

The Internet of Things (IoT) could change the agriculture industry by making it possible to build smart, automated control systems that can maximise crop yield while using the least amount of energy. A greenhouse equipped with an intelligent control system, aided by IoT technology, has better energy-saving capabilities. IoT is a concept that supports the connection of physical devices exchanging data over the internet for monitoring and control purposes (Markovic et al., 2016)(Markovic et al., 2016) . Through an intelligent control system aided by the IoT, a greenhouse can save more energy and reduce emissions.

Despite the potential benefits of a smart control system (or monitoring system as addressed in some writings) for greenhouse management, its effective implementation in greenhouse agriculture remains a significant challenge (H. Li et al., 2021). In commercial greenhouses, control strategies such as On-Off (bang-bang) control, proportional integral derivative (PID), feedforward control, fuzzy control, model predictive control (MPC)**,** and optimal control have been utilised to maintain optimal growing environments and to optimise plant growth. By improving these established control strategies, the sensors and actuators of the control system can more efficiently maintain the desired growing conditions for plants, ultimately leading to energy savings and efficient operation. The need, therefore, is to improve the performance and efficiency of these established strategies. In particular, optimizing the performance of On-Off control is a novel approach that can contribute to energy savings.

1.1 Aim

To design and implement an IoT-enabled greenhouse automatic control system for optimising energy efficiency in greenhouse.

1.2 Research questions and objectives

Greenhouses are a major energy-consuming sector in the agricultural industry (Iddio et al., 2020a). An efficient control strategy that can adjust and coordinate equipment for low-energy operation is a promising way to reduce total energy consumption in greenhouses (S. Zhang et al., 2020a). Reduced greenhouse energy consumption is beneficial to both growers and society. While IoT has

been deployed in various application domains such as transportation, healthcare, smart homes, schools, and markets, its use in energy minimization in agriculture is still scarce. As agricultural practices increasingly use more advanced equipment, incorporating advanced control strategies to reduce energy consumption will become more urgent (van Beveren et al., 2015). This research aims to answer the following question:

How can an automated control system based on Internet of Things (IoT) technology optimize energy use in greenhouse systems?

Despite the general adoption of IoT technology in various agricultural applications, integrating numerous heterogeneous devices aided by IoT over the internet to minimize energy use is an important challenge that is hardly considered.

Objectives

- To modify an existing physics-based greenhouse temperature model to account for additional heaters and simplify the model through optimisation for controller design.
- To evaluate the performance of two groups of controllers (On-off and PI controllers; On-off, PI, PID, and MPC controllers) in achieving the user's desired setpoint and reducing energy consumption in the greenhouse, based on the modified physics-based model.
- To design and implement an IoT-based control system for monitoring and controlling greenhouse temperature and to deploy it in a pilot greenhouse system.

1.3 Structure of the thesis

This work is organised as follows in light of the preceding: The first chapter provides an overview of the greenhouse industry as well as the significance of energy management in greenhouse operations. In addition, it outlines the research questions and objectives of the thesis and provides a summary of the structure of the thesis.

Chapter 2 of the thesis is the literature review. This chapter discusses the various strategies that have been implemented to manage energy consumption in greenhouse operations. It offers an overview of the greenhouse energy simulation models and controller algorithms used in the work, as well as a literature review on IoT technology and its application in greenhouse operations. It also discusses the various types of sensors and actuators used in greenhouse operations as well as their functions in energy management.

Chapter 3 introduces modelling and its significance for energy management. It first discusses the physics-based and First-Order Plus Dead Time (FOPDT) models used in the research. Then, it presents the design of the simplified physics-based model through optimisation and the parameter estimation techniques for greenhouse temperature control. Finally, the chapter summarizes the findings and discusses the implications of the study.

Chapter 4 discusses the On-off, PI, PID, and MPC controllers used in the research. In addition, the chapter presents the implementation of the PI and On-off controllers based on the physics-based temperature model for maintaining the growers' desired setpoint, as well as the implementation of the On-off, PI, PID, and MPC controllers for energy minimization capability. The chapter applies IMC tuning to optimise the performance of the controllers, compares their performance, summarises the findings, and discusses the implications of the study.

Chapter 5 describes the design and implementation of the IoT control system for greenhouse energy management. It outlines the procedures followed for the software and hardware implementation and also discusses the materials and database services used in the system. Additionally, the chapter presents an evaluation of the control system's performance and concludes by discussing the results obtained.

Chapter 6 presents the experimental setup and methodology used in the study, including the selection of materials and equipment, the measurement of relevant variables, and the design of the experiments. The chapter also describes the collection and analysis of data, including the statistical methods used to interpret

the results. The study's findings are then presented, with a particular focus on the performance evaluation of the PID controller. The results are summarised and discussed in the context of the research questions and objectives.

Chapter 7 provides a summary of the research questions and objectives, a summary of the major findings and contributions of the study, a discussion of the study's limitations and suggestions for future research directions, as well as concluding remarks and suggestions for additional research.

CHAPTER 2 Literature review

2.1 Energy challenges faced in the greenhouse industry

As the population of the world rises, so does the demand for food. Unfortunately, climate conditions can have a negative impact on outdoor crop production. Agricultural greenhouse systems provide a solution by capturing short wavelengths of solar radiation and storing long wavelengths of thermal radiation to produce a microclimate conducive to increased crop yield. These buildings are also known as controlled environments (Jain & Tiwari, 2003).

Over 2000 years ago, the Romans discovered that crops could be protected from adverse environmental conditions by using light-transmitting shelters, which enabled the growth of cucumbers during winter and spring (STANGHELLINI, n.d.). They also realized that productivity could be improved by actively adjusting the climate with heating and humidification. However, due to limited knowledge of crop productivity and growth processes, and the unavailability of equipment for climate conditioning, advanced climate control strategies were not implemented (Henten, 1994).

As technology advanced over the years, greenhouse cultivation became the most developed field of horticulture. The availability of technological advancements and the increased demand for luxury crops out-of-season, due to the growing wealth of people, contributed to this development. Today, thanks to changes in production handling and an increase in the scale of crop production, fresh greenhouse products are available year-round to the world.

The barrier between the crop and the outside environment is a unique feature of greenhouse cultivation that protects crops from weeds, wind, pests, precipitation, animals, and diseases. It creates a different micro-climate inside the greenhouse and enables farmers to incorporate environmental control systems that differ from outdoor farming. This protection from outdoor environmental conditions also allows for the effective application of chemical and biological control, heating, and the addition of carbon dioxide. Due to the higher return generated in profit per unit area of produce, growers are motivated to invest in sophisticated equipment

such as supplemental lighting, cooling systems, soil cooling/heating systems, etc. Greenhouse cultivation is now considered the most sophisticated and intensive method of crop production, emphasising the role of technology in the entire process (Bakker, J.C., Bot, G.P.A., Challa, 1995).

The agricultural production of a nation largely determines its food security. However, it is very sensitive to several factors, such as climatic conditions, water resources, and energy resource management. To expand the agricultural sector, it is essential to introduce new and novel farming technologies that emphasise economic efficiency, minimise environmental impact, and optimise the use of scarce resources. One crucial aspect of this strategy is the adoption of protected and controlled environments, particularly greenhouse systems. In the last few decades, greenhouse crop production has increased worldwide. However, optimal plant yield requires that indoor conditions be controlled and determined (Maher et al., 2016),

The basic energy consumption of a greenhouse includes the energy used for light supplementation, dehumidification, heating, cooling, and other measures. The energy needed to power the actuators is another part of energy usage. The greenhouse's basic energy consumption could account for more than 90% of its total energy consumption (Shen et al., 2018).

According to (Ahamed et al., 2019), in cold climates, heating represents 65-85% of the total energy used in greenhouses, with transportation and power accounting for the remaining energy usage. They further emphasise that, excluding labour costs, heating (mostly) and cooling costs make up 70-85% of overall operational expenses. In warmer regions like the Southwest US, Mexico, Spain, Israel, etc., costs might still be 50% of overall operating costs (Rorabaugh, Jensen, and Giacomelli, 2002). The high energy costs associated with heating pose a significant challenge to greenhouse operators. Therefore, reducing greenhouse heating costs could make greenhouse production more economically and environmentally sustainable.

2.2 Previous methodologies used to conserve heating energy

To reduce heating energy consumption, several methods have been implemented, including passive solar design, energy-efficient glazing, thermal insulation, and ventilation control. Passive solar design involves the use of building materials and design elements to capture and retain solar radiation, such as selecting glazing with high solar transmittance and low thermal transmittance. Energy-efficient glazing, on the other hand, involves the use of double or triple glazing, low-emissivity coatings, and other technologies to reduce heat loss through the glazing.

Before further discussion of energy-saving methods for commercial greenhouses, it is necessary to examine the greenhouse's heat gains and losses first. It is crucial to understand that the greenhouse operates on the basis of the greenhouse effect, and it is necessary to study the principle of heat gains and losses in greenhouses to comprehend how the thermal processes operate. Figure 2.1 (Vadiee, 2013) illustrates how short-wavelength solar radiation (visible light) passes through the glazing of a greenhouse and is absorbed by objects inside.

These objects then emit long-wavelength infrared radiation that cannot fully pass through the glazing, causing an accumulation of heat and resulting in an increase in indoor temperature. Additionally, plants transpire CO_2, which is an effective infrared radiation absorber, leading to higher concentrations of CO_2 inside the greenhouse and further warming the space.

While radiation is the primary source of heat gain in a greenhouse, the long-wavelength radiation that travels through the glazing also has a significant impact on heat loss at night. As the components of the greenhouse are typically warmer than the surrounding ambient temperature, they emit heat as infrared radiation through a process called radiation cooling. Apart from radiation, highly

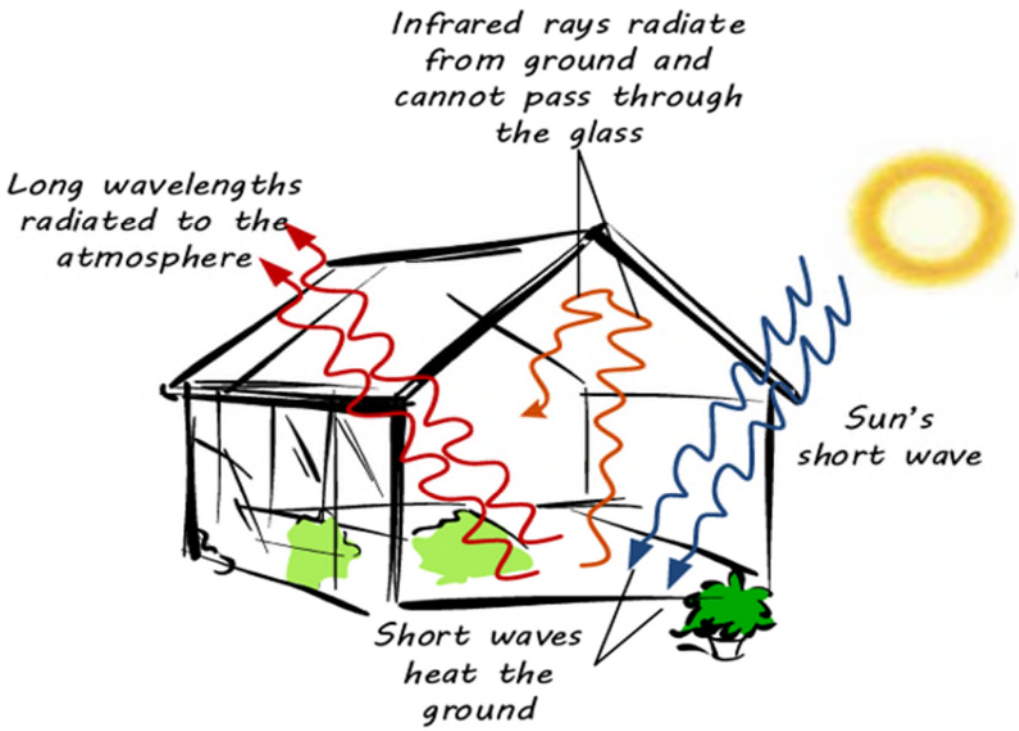


Figure 2.1 Radiation Energy flow in a greenhouse

efficient heating systems such as heat pumps, boilers (Tunc et al., 1985), electric space heaters, etc., and artificial lighting (Serale et al., 2021), which also emit heat, are sources of heat gain. Along with these heat gain sources; conduction and convection also contribute to heat gain and loss in a greenhouse.

Figure 2.2 (Sethi et al., 2013) below shows that conduction heat loss occurs mainly through the floor and the soil, and equally through the glazing, while ventilation (natural and forced) and infiltration are the primary sources of convection heat transfer.

Other significant convective heat fluxes in the greenhouse include convective heat flux from the indoor air to the crop, soil, and roof; these must be accounted for in the greenhouse energy balance (Vadiee, 2013). With the knowledge of heat gain and loss, it can be concluded that the main sources of heat gain are the radiation from the sun, heating systems, and artificial lighting. While the main sources of heat loss are radiation cooling, leakages in the structure, and

ventilation, understanding these sources and losses can help identify effective ways to reduce energy consumption and save costs.

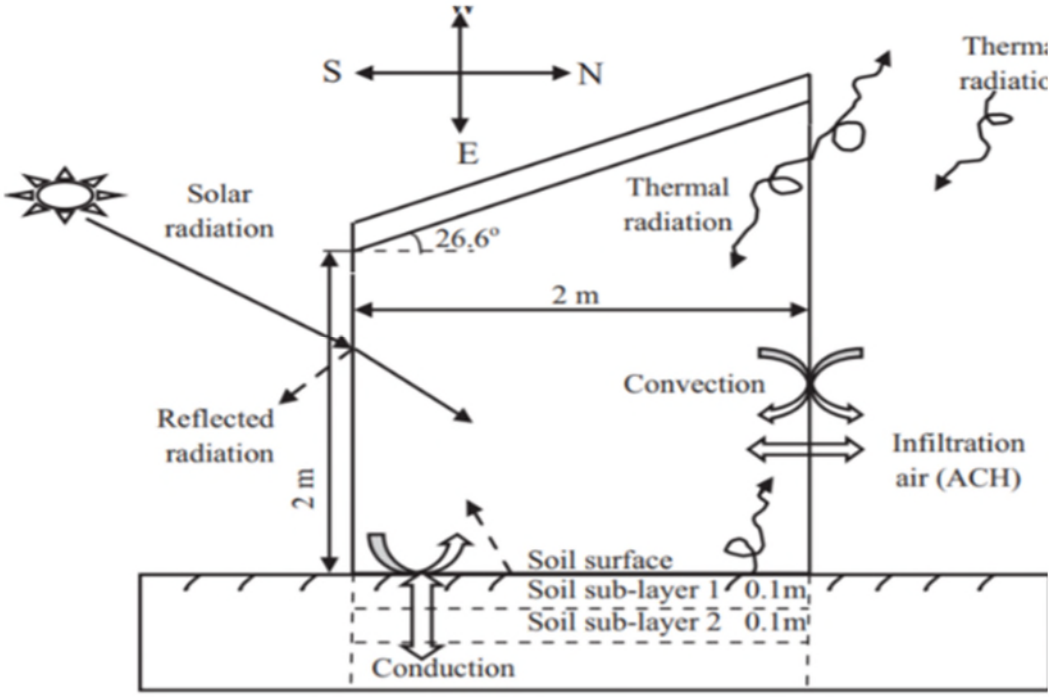


Figure 2.2 Heat exchange with the surrounding

In his paper, Vadiee summarizes the energy-saving methods evaluated by Nederhoff and Bartok in "Improving Energy Efficiency in Greenhouse Vegetable Production" (as cited in "Energy Management in Large Scale Solar Buildings: The Closed Greenhouse Concept") and categorizes them into three groups, as shown in Table 2.1 (Vadiee, 2013).

From Table 2.1 below, the options presented within the low impact energy saving category offer a potential reduction in energy consumption ranging from 1% to 2% within the context of a commercial greenhouse.

Meanwhile, the implementation of the medium impact energy saving (second group) category can lead to a potential increase in energy savings of up to 5%. It is worth noting that natural ventilation can aid in the dissipation of excess heat and moisture through exchange between the inside and outside air ((Singh et al., 2018). reducing the need for artificial heating and cooling. However, growers

must also heat the greenhouse to compensate for heat loss through ventilation windows, resulting in additional energy costs (Vadiee, 2013).

Furthermore, implementing any alternatives categorized under the high impact energy saving (third group) category can result in an additional 5% improvement in energy conservation. While certain individual solutions may have a moderate or minimal impact, they can still result in significant energy conservation. For instance, better insulation and high-performance glazing systems in the construction of the greenhouse are identified as parts of the high impact energy saving solutions in Table 2.1 (Vadiee, 2013).

Table 2.1 Different means for energy performance improvement in a greenhouse

Low impact (1-2%)
• Using variable speed or frequency-controlled pumps • Pass the heating pipes along the greenhouse wall with sufficient space between them • Using insulation into the boiler house's walls
Medium impact (3-5%)
• Calibrate and regulate the ventilation windows hydraulic system • Keep the ventilation windows and fans closed/shut off while they are not used • Ensure regular system maintenance • Repair the gaps and cracks in the greenhouse structure • Insulate the greenhouse foundation and sidewall • Minimize the light blocking object above the crop • Install the heating pipes as close to the cultivated area as possible • Insulate the heating and cooling pipes and ducts
High impact (More than 5%)
• Using highly accurate climate control system • Using the heating system with high efficiency (higher performance) • Using highly efficient artificial lighting system • Optimizing the CO2 enrichment control system • Using better insulation in the greenhouse construction • High performance glazing system • Using thermal screen • Using short and long term thermal storage system • Regular maintenance of the weather station

However, maintaining the required inner environment with these materials is challenging due to their transparency (Liu et al., 2016). Additionally, energy-efficient LED lights and adjusted lighting schedules can minimize energy consumption, while higher performance heating systems such as high-efficiency

boilers, efficient heat pumps, and efficient space heaters can also reduce energy use.

To increase the impact of energy-saving methods in different categories, highly accurate automatic control systems can be integrated with one or more solutions from the different groups. For example, a smart automatic control system that is event-based can carefully consider control actions in relation to disturbances in the surrounding elements, making it more efficient. Interestingly, this approach to energy savings is not widely explored in the literature.

In addition to the methods summarised by (Vadiee, 2013), another prevailing energy-saving approach is the use of various energy storage equipment (Van Beveren et al., 2013). These systems, such as aquifers, buffers, and heat exchangers, are used in closed and semi-closed greenhouses. However, controlling these systems can be challenging due to different types of energy and complex facilities, among other factors (Singh et al., 2018).

Automated control systems use sensors to monitor temperature, humidity, light levels, and other factors and adjust heating, cooling, and ventilation systems accordingly to optimise resources in the greenhouse. Automating greenhouse processes using technologies such as information and communication technology (ICT) is an efficient way of minimising resources. The greenhouse agriculture industry is increasingly investing in technologies, transitioning from traditional to ICT-controlled systems, resulting in a smart greenhouse environment (Bersani et al., 2021) or precision farming PA.

ICT has been implemented in the greenhouse industry for diverse applications. With the incorporation of different sensors and actuators, automated farms can monitor the environment and make appropriate decisions (X. Jiang, 2020). The use of ICT in building automated control systems can help monitor the environment and make smart decisions to optimize energy use and increase economic efficiency while minimizing the impact on the environment. In fact, PA relies on ICT solutions to monitor, measure, and control internal parameters in order to boost efficiency by limiting environmental effects, permitting cost and water savings, etc. (Bersani et al., 2021). IoT is a fast-growing technology that is

lately adopted to automate and remotely control the environmental parameters in greenhouses using hardware resources (sensors, controllers, and actuators) and control algorithms.

For instance, the use of better insulation in the construction of greenhouses from Table 2.1 can be merged with the IoT-based Arduino control system to create a more efficient heating and cooling system. Better insulation will help to reduce heat loss and maintain a more stable temperature inside the greenhouse, which can help to reduce the workload of the heating and cooling systems. The IoT-based Arduino control system can be used to monitor the temperature inside the greenhouse and adjust the heating and cooling systems accordingly. By using the automatic control system to maintain a more stable temperature, the insulation can even be more effective in reducing energy consumption. In addition, the control system can also be used to monitor and control the heating and cooling pipes and ducts mentioned in the medium impact energy saving of Table 2.1**,** ensuring that they are working efficiently and not wasting energy. By merging better insulation with IoT-based Arduino control system, a more efficient and cost-effective heating system for a greenhouse can be created.

Greenhouses are complex environments that require precise control of various parameters to ensure optimal plant growth and productivity. Automatic control systems can help achieve this by utilising hardware resources such as sensors, controllers, and actuators, along with control algorithms, to monitor and adjust the greenhouse environment. The use of IoT hardware resources has been explored in various greenhouse applications, including optimal control of soil nutrient levels in hydroponics (Zamora-Izquierdo et al., 2019), intelligent lighting control systems (Afzali et al., 2021; J. Jiang & Moallem, 2020), optimisation of greenhouse irrigation systems (Navarro-Hellín et al., 2016; Vineela et al., 2018), and real-time monitoring of crop-sensitive data (Thomopoulos et al., 2021). However, the literature is scarce when it comes to reducing greenhouses' total heating energy demand using IoT hardware resources. Only a few studies have explored this topic (Sagheer et al., 2021; Šaš & Pejić, 2021), and they did not use greenhouse energy models to implement the IoT-based automatic control system solution.

This makes it challenging to determine if total energy usage is reduced in these applications since all the fluxes that influence the greenhouse climate were not accounted for in simulation and control system design. Therefore, there is a need for further research in this area to develop effective IoT-based control systems that can reduce energy consumption in greenhouses while maintaining optimal growing conditions for plants.

2.3 Overview of greenhouse dynamics energy simulation model

A greenhouse energy simulation model is a computational tool used to predict the interior climate and estimate the energy consumption of a greenhouse. The model may consider various factors, such as greenhouse design, plant growth, weather conditions, and heating and cooling systems. By simulating the greenhouse environment, the model can provide insights into the dynamics of the system and help optimise energy management strategies. The use of modelling tools can improve the understanding of greenhouse operations, reduce energy costs, and promote sustainable farming practises.

Since Businger's development of an energy budget model for greenhouses in 1963, greenhouse modelling has undergone significant growth, as reported by Katzin et al. Subsequently, greenhouse modelling has been used for various purposes, including in research for the synthesis and advancement of knowledge, as an educational device in the classroom, and as an aid to decision-making and policy analysis (Katzin et al., 2022a). Over time, the progress in modelling has led to the development of models for both single and multiple component systems, and they have been built using both static and dynamic approaches. Analytical and numerical models (HWANG, 1993), as well as the quasi-steady-state model (Shamim, 2018), have also been designed. These different models serve different applications, from studying thermal exchange phenomena to designing and controlling greenhouses (HWANG, 1993). The proliferation of greenhouse models may be attributed to the extreme versatility of greenhouses, which differ in structure type, climate control equipment, cultivated crops, and intended uses (Katzin et al., 2022a).

There are three methods for computing a model: physics-based modelling, empirical modelling, and hybrid modelling. Physics-based modelling is based on the physical laws involved in the process and uses the thermodynamic properties of the system. Empirical modelling, on the other hand, analyses input and output data from the process to infer a model (Boaventura Cunha et al., 1997), which is also known as the data-driven modelling approach. The hybrid model combines the physics-based and data-driven modelling approaches and is useful when there is a lack of data or incomplete physics knowledge of the system. It consists of a state-space model generated using the resistor-capacitor (RC) model, which describes thermal dynamics through energy balance. The model is constructed as a first-order differential equation of the system states, disturbances, inputs, and outputs. The model parameters are estimated, and the data-driven approach is used to train the model (Iddio et al., 2020a).

The advantages of empirical models like the FOPDT model are that they are relatively easier and faster to develop than physics-based models and can be effective when there is not enough understanding of the underlying physics or when a large amount of data is available. However, they are generally less accurate than physics-based models when extrapolating beyond the range of data used for calibration and may lack generalisation, meaning they may not be applicable to situations beyond those used for calibration. Additionally, empirical models often provide little insight into the underlying physical mechanisms of the system, which can limit their usefulness in certain applications.

Physics-based models, on the other hand, are more accurate and reliable than empirical models when the underlying physical mechanisms are well understood. They can provide a better understanding of the underlying physical mechanisms and allow for the testing of hypotheses and scenarios. They can be used to make predictions beyond the range of data used for calibration. However, they can be computationally expensive and time-consuming to develop, require a good understanding of the underlying physical mechanisms, which can be challenging to obtain for complex systems, and are often difficult to calibrate. The uncertainty in model parameters can also affect model accuracy.

The advantages of hybrid models are that they can provide a balance between accuracy and computational efficiency by combining empirical and physics-based approaches. They can incorporate the strengths of both empirical and physics-based models to provide more accurate predictions, and they can allow for easier calibration and more straightforward use than pure physics-based models. However, they can be challenging to develop and may require significant computational resources. Issues can arise with the integration of the empirical and physics-based components, leading to difficulties in calibration and accuracy.

The traditional method for modelling variables such as temperature, humidity, and CO2 concentration are to consider the balances: an energy balance, a CO2 balance, and a water vapour balance. However, not all greenhouse climate models in the literature describe all three balances. Some focus only on the energy balance, while others consider both the energy and water vapour balances. These balances identify the incoming and outgoing flows of each variable, and the differences between these flows determine the net change in the variable (Katzin et al., 2022a). Temperature, light, humidity, and CO2 concentration are some of the variables that can be used to describe the indoor greenhouse climate.

Researchers have developed models for greenhouse systems to optimise energy use. To model a greenhouse system, it is common to divide it into multiple components, as the thermal interactions between these components can be intricate. Typically, greenhouses are divided into five or fewer components, including the internal air, soil surface, transparent cover, plant, and soil sub-layers. Researchers employed energy balance equations to create models that were suited to their individual application needs. Sub-models for particular greenhouse components have been developed utilising various heat transfer methods, including radiation, conduction, convection, and ventilation, as mentioned in Section 2.2. These sub-models are then combined to form an overall model to meet the application's requirements.

This section aims to identify the most appropriate greenhouse model for energy optimisation when using IoT-supported hardware. To achieve this, existing

models in the literature are reviewed and classified into two groups: dynamic greenhouse models and dynamic energy models. The primary difference between these categories is their focus. Dynamic greenhouse models concentrate on simulating and predicting the behaviour of greenhouse systems, such as temperature and humidity, the growth and yield of crops, and external weather conditions. While they may include energy-related variables, their primary focus is on optimising crop growth and yield while maintaining a stable environment inside the greenhouse. On the other hand, dynamic energy models focus on optimising energy flows and exchanges within greenhouse systems, such as heating, cooling, lighting, and ventilation, while maintaining a comfortable indoor environment. Some models in this category may also consider plant growth and yield as secondary objectives. The review of these models will help identify the most appropriate model for energy optimisation using IoT-supported hardware in the greenhouse.

2.3.1 A Review of dynamic greenhouse models

A greenhouse climate model, whether static or dynamic, can be utilised to simulate temperature, solar radiation, humidity, and carbon dioxide (CO2) and can be used to control equipment fluxes by employing available weather conditions, as discussed previously. However, a dynamic model of the greenhouse, in particular, can simulate the time-varying control behaviour (HWANG, 1993) or the effect of the system input variables on the output variable. Knowing this effect is crucial to building a controller that can efficiently manipulate the output variable to maintain the desired temperature value.

Several studies have been conducted to develop a greenhouse dynamics model that describes heat and mass transport processes (J. Chen et al., 2016; Soribe & Curry, 1973; Tiwari et al., 1998). It is important to note that models are designed based on the researcher's needs, which also affect the components included in the model's design.

Abdel-Ghany and Kozai (Abdel-Ghany & Kozai, 2006) developed a dynamic simulation model for heat and water vapour transfer in a naturally ventilated, fog-cooled greenhouse to predict air, plant, cover, and floor surface temperatures, as

well as relative humidity inside the greenhouse. The model also predicted transpiration and evaporation. The model was based on solving a system of equations numerically using the predictor-corrector technique for differential equations and the iteration procedure for algebraic equations. The input parameters to the model included meteorological conditions and the thermophysical properties of the greenhouse cover, plants, air, and soil.

An experiment was conducted to measure the environmental conditions inside and outside the greenhouse, which was intermittently cooled by spraying water fog at different rates and intervals. The predicted results using the model were compared with the measured values, which showed good agreement at different fogging and interval times. Therefore, the model was suggested as a useful tool for predicting indoor temperatures in naturally ventilated, fog-cooled greenhouses. This model falls under the category of a "greenhouse dynamic model" due to its focus on predicting temperatures and humidity inside the greenhouse rather than energy optimization.

The work by Joudi and Farhan (Joudi & Farhan, 2015) aimed to integrate a conventional greenhouse with roof-mounted solar air heaters to reduce the load and cost of cooling while providing solar heating. They developed a dynamic model that predicts the internal temperatures of the greenhouse, taking into consideration soil surface heat exchange with the greenhouse air. This feature improved the accuracy of temperature prediction, and the study identified the best location for heat storage elements. On the other hand, Fitz-Rodríguez et al. (Fitz-Rodríguez et al., 2010) developed an interactive greenhouse environment simulator based on energy and mass balance principles. The simulator allows users to select different greenhouse designs, weather conditions, and operational strategies to understand the dynamic behaviour of greenhouse environments. Although energy and mass balance principles were incorporated into the simulator, its primary focus is on simulating the greenhouse environment, rather than optimising energy use.

The focus of Imen Haj Hamad et al.'s study was to develop a nonlinear neural autoregressive with exogenous inputs (NNARX) system for modelling the internal

temperature of a greenhouse as a function of various environmental factors such as outside air temperature and humidity, global solar radiation, and sky cloudiness. The study found that the balance between the number of neurons in the hidden layer of the NNARX system and the number of iterations for model training is critical to achieving good performance. Their model demonstrated good performance over extended periods without frequent parameter tuning, which is essential for dynamic greenhouse models (Hamad et al., 2021).

In contrast, J. Boaventura Cunha et al.'s work aimed to achieve efficient real-time control of the greenhouse environment through the implementation of a recursive identification technique. Their study focused on real-time estimation and identification of parameters of a climate dynamic model for a greenhouse, with the aim of developing a discrete-time dynamic model of greenhouse air temperature based on experimental data from two different periods of the year. The study describes the methods used to estimate the parameters of this model in real-time, considering the fact that the parameters may vary with different operational conditions. The implemented recursive identification technique for parameter estimation enables more efficient use of the model in real-time control. This work falls under the category of dynamic greenhouse models, as discussed earlier, as it does not focus on energy optimisation (Boaventura Cunha et al., 1997).

2.3.2 Dynamic energy models aimed at energy optimization.

The use of a dynamic air temperature model of a greenhouse is essential to describing the climate inside the prototype greenhouse (that was used in this work) and the functionality of the control equipment. This is important because, to control the greenhouse, a model-based control design is necessary to understand the behaviour of the system and make predictions for control purposes (Speetjens et al., 2010). Therefore, a dynamic energy simulation model helps to simulate the temperature dynamics, facilitate understanding of the complex mechanisms involved in the thermal process of greenhouse operation, and thus contribute to a more energy-efficient greenhouse operation. The model that is most appropriate for this work is the one designed for the control of

greenhouse equipment for the optimisation of energy resources and enabling the calculation of the required total energy input to achieve the pre-set greenhouse climate.

The work done by Dilip Jain and G.N. Tiwari involves the use of a mathematical model to study the thermal behaviour of a greenhouse using a ground air collector. The model was developed using Matlab software and was validated experimentally. The author’s work is to predict the plant and room temperatures as a function of various design parameters of the ground air collector, such as area, mass flow rate, and heat capacity, to optimise the greenhouse's thermal performance. Although their work does contribute to a better understanding of the thermal dynamics of greenhouse operation and supports more energy-efficient operation, it does not focus on controller design (Jain & Tiwari, 2003).

Gutman et al.'s (1993) developed a Model Predictive Control for greenhouse operation by solving a new optimal control problem each time the weather prediction was updated. Their dynamic energy model is used for controlling the climate inside a greenhouse with a simplified nonlinear dynamic model of greenhouse crop growth that has constraints on the state and control signals. The optimisation criterion was to minimise heating costs, and the solution was found using linear programming and network flow problem formulation. The study showed that deviating from the "blueprint" and heating during the night when the thermal screens were in place and the heat loss was small could lower heating costs by 22% relative to a constant temperature operation. Their optimisation criterion though targeted at minimising the cost of total energy input, however, it includes an economic criterion (GUTMAN et al., n.d.)**.**

The study by Van Ooteghem et al. presents a solar greenhouse that utilises renewable energy sources to minimise fossil fuel consumption. Their greenhouse is equipped with a heat pump, heat exchanger, aquifer, and ventilation with heat recovery to regulate temperature and humidity within certain limits. Their objective is to minimise fossil energy consumption while maximising crop dry weight and maintaining a temperature integral. To achieve this, they use the receding horizon optimal controller strategy, which computes optimal closed-loop

controls based on a cost function and a greenhouse and crop model. Through simulations of a simplified version of this closed-loop optimal control system, they found that boiler use is reduced, thereby reducing fossil energy use. The use of gas is decreased by 77% compared to a conventional greenhouse with optimal control. Their cost function also includes economic factors such as energy costs and crop yield values (Van Ooteghem et al., 2005).

Although dynamic greenhouse models have been developed to predict variables in the greenhouse environment, they are not the preferred models in this work, as demonstrated in the reviewed studies in Section 2.3.1 and the works above. Abdel-Ghany and Kozai developed a greenhouse model to predict the temperatures and humidity in a naturally ventilated, fog-cooled greenhouse, while Joudi and Farhan developed a dynamic model to predict internal temperatures and investigate the contribution of soil surface heat exchange. Fitz-Rodrguez et al. developed an interactive greenhouse environment simulator for educational purposes, and Imen Haj Hamad et al. developed an NNARX system for predicting greenhouse temperatures based on environmental factors. J. Boaventura Cunha et al. developed a model that estimates the parameters of a greenhouse dynamic model in real-time. However, these models do not directly target energy optimisation.

Dynamic energy models are commonly used to minimise energy input while maintaining a controlled environment. These models include the work of Gutman et al., who developed a dynamic energy model to minimise heating expenses, and the study of Dilip Jain and G.N. Tiwari who developed a mathematical model to study the thermal behaviour of a greenhouse using a ground air collector to optimize its thermal performance. Van Ooteghem et al. designed a solar greenhouse with a receding horizon optimal controller to minimise fossil energy consumption while maximising crop dry weight and temperature integral and found a 77% reduction in gas use compared to conventional greenhouses.

While Gutman et al. and Van Ooteghem et al. focused on optimal control strategies with an economic criterion to minimise heating costs in simulation models, they are not practical due to a lack of trustworthy crop production models

for the diverse array of crops and species cultivated in horticulture practise and the requirement to delegate certain decision-making responsibilities to growers (Van Beveren et al., 2013). Dilip Jain and G.N.'s work supported energy-efficient operation without a direct focus on optimisation through a controller. However, for an IoT based control system, a physics-based model that directly addresses the minimization of total input energy use is more appropriate.

As far as is known, Chalabi et al. have reduced total energy consumption without using an economic criterion, but they used a steady-state temperature model. They present a real-time control algorithm for optimal heating setpoint generation in a commercial greenhouse with a tomato crop. The algorithm is based on a numerical optimisation method and a greenhouse energy requirement model that uses weather forecasts from the Meteorological Office. The overall system is designed to require minimal additional intervention from the grower, who only needs to define the constraints on the heating setpoints (Chalabi et al., 1996).

Van Beveren et al. developed and validated the dynamic air temperature model, and they used optimal control strategies to reduce the greenhouse's overall energy input. It was found that when cooling is applied, the air temperature is on the upper boundary, and when heating is applied, it is on the lower boundary (Van Beveren et al., 2013). This study is considered the most relevant for the current research because it includes an energy module that enables the implementation of a controller design suitable for IoT applications. The present work improves upon their physics-based air temperature model and uses an optimisation-based model fitting technique to adapt the model for simulating a prototype greenhouse equipped with IoT hardware resources to predict indoor air temperature. The modified model is validated using data from the built IoT cloud-based control system deployed in the greenhouse, and the discussion on the model is presented in Chapter 3.

2.3.3 Control theory and algorithm for greenhouse systems

Recent literature has shown an increasing interest in using advanced control algorithms, such as model predictive control (MPC), to optimise greenhouse

energy use. However, there is also a need for simple and low-cost control algorithms that can be used by the average farmer. One such strategy is on-off control, which is rarely considered in the literature as a control strategy for energy optimisation. This study aims to assess the suitability of on-off control by first comparing its performance to that of the PI control algorithm using the optimised physics-based model developed for maintaining the grower's desired indoor temperature. Secondly, the study aims to compare its performance in simulation for energy reduction with PI, PID, and MPC control strategies, using the optimised physics-based model.

Controllers are essential devices for optimising the energy consumption of a greenhouse system by receiving input signals and producing output signals to control processes and systems. In greenhouse energy optimisation, controllers play a critical role in monitoring environmental variables such as temperature and humidity and adjusting equipment and systems to maintain desired conditions while minimising energy use. As growers increasingly seek to reduce energy consumption and minimise their environmental impact, the use of controllers in greenhouse systems has become more important. Automation and optimisation through controllers have resulted in significant improvements in crop efficiency, yield, and quality. This section provides an overview of control theory, a brief discussion of different greenhouse control concepts, and a description of the four controllers used in this study, including On-off, proportional-integral-derivative (PID), and model predictive control (MPC).

Control theory is a field of science that deals with influencing the behaviour of dynamic systems. Its goal is to ensure that one or more system output variables follow a particular reference over time. Controllers achieve this by acting on the system inputs to obtain the desired effect on the outputs.

One basic approach for controlling a system is open loop control, as shown in Figure. 2.3 (Romero et al., 2012). In this mode, no measurements from the system outputs are used to modify the inputs, which means that no feedback is used.

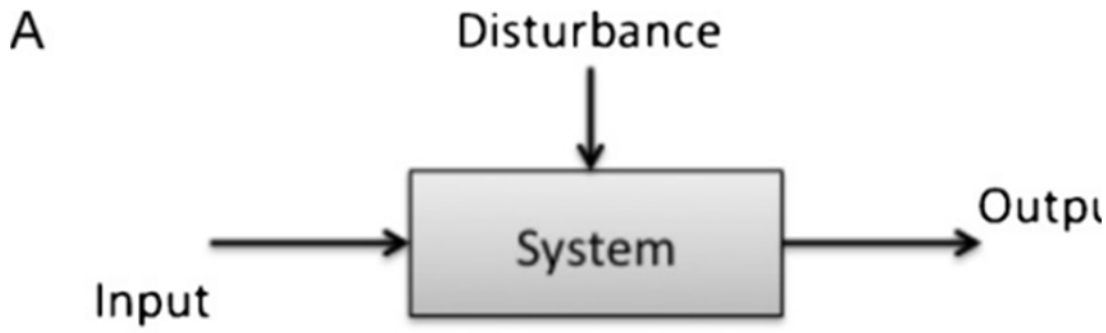


Figure 2.3 Open loop controller

The decision in this class of controllers is governed based on heuristics, expert knowledge, or the model of the system.

Another control theory used in the control of dynamic systems is the feed-forward strategy, as shown in Figure. 2.4. This approach is based on using known or estimated values of future disturbances to compensate for their effects in advance before they affect the system output. One of the main advantages of this approach is that it can prevent or reduce the impact of disturbances before they occur, leading to improved system performance. However, a key drawback is that feed-forward controllers are not able to react to changes in the actual disturbances or in the system, which can limit their effectiveness in some scenarios.

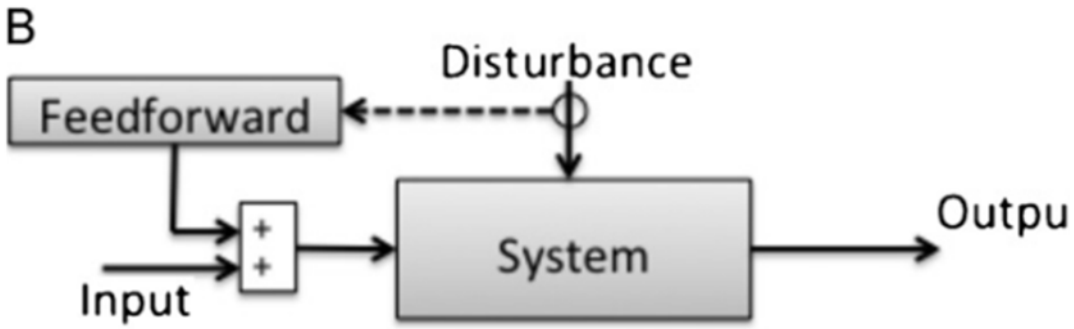


Figure 2.4 Feed-forward controller

In contrast, closed-loop controllers, as shown in Figure. 2.5, use feedback to mitigate the inadequacies of open-loop controllers. In this approach, the controller uses information from sensors that measure the system output to adjust the input signal and achieve the desired system response. By using feedback, closed-loop controllers can compensate for disturbances and system uncertainties, leading to improved control performance and stability.

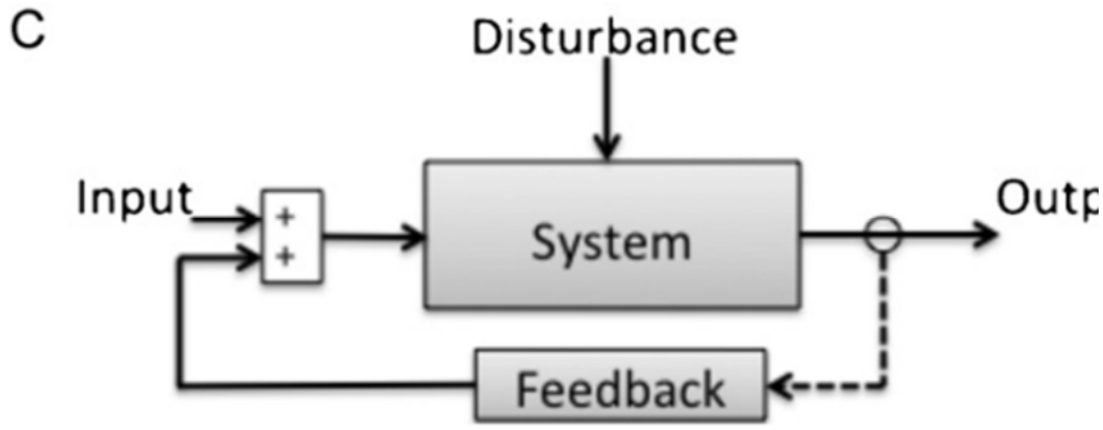


Figure 2.5 Feedback or closed loop controller

However, closed-loop controllers require appropriate sensors to measure the system output and close the control loop, which can increase the complexity and cost of the control system.

2.4 Greenhouse control algorithm

Several control algorithms have been developed over the years to reduce greenhouse energy consumption. In this work, the focus is on the four control algorithms: On-off, PI, PID, and MPC.

2.4.1 On-off control

The On-off controller is a simple type of controller which switches the control signal on and off based on a fixed set point. The On-off controller is commonly used in heating and cooling systems to maintain the temperature within a certain range. When the temperature drops below the set point, the controller switches on the heating system, and when the temperature rises above the set point, it switches off the heating system. The On-off controller is known for its simplicity, low cost, and ease of implementation. and it may also be used as a substitute for a PID controller, particularly in temperature control. A thermostat is a typical example of an On-off controller that is usually used in room temperature control (Haugen, 2021).

However, it has some limitations. One of its major limitations is its hysteresis effect, which can lead to temperature oscillations around the set point. Additionally, the On-off controller does not provide precise control and cannot be used for systems that require precise temperature control (Haugen, 2021). The On-off controller function is given in (Haugen, 2021)**:**

$$u_{(t)} = \begin{cases} u_{max} & if\ e\ \geq 0 \\ u_{min} & if\ e < 0 \end{cases} \qquad 2.1$$

Where;

$$e\ = r(t) - y(t) \qquad 2.2$$

$\mathrm{e}(\mathrm{t})$ is the control error, $r(t)$ is the setpoint or reference signal and $y(t)$ is the process output measurement, u_{max} is the maximum control signal, usually 100%, and u_{min} is usually 0%. The increased attention paid to the use of advanced control algorithms, such as MPC, for the optimisation of greenhouse energy use is noted in the literature (Iddio et al., 2020b; S. Zhang et al., 2020b), and it has

been suggested that there is a need for a straightforward, inexpensive control algorithm that can be used by the average farmer. ON/OFF control is a simple and affordable temperature control strategy that is rarely considered for reducing energy use in greenhouses. To assess its suitability for minimising energy consumption, this study compared its performance to that of the PI, PID, and MPC algorithms in energy optimisation. Sagheer et al used On-off control in their work on controlling soilless greenhouse cultivation using an IoT precision control system inside a commercial-sized greenhouse for the regulation of different greenhouse environmental parameters; however, it is hard to determine if the total energy used was reduced in their work because all the fluxes that affected the greenhouse climate were not accounted for in their simulation and control system design (Sagheer et al., 2021).

2.4.2 Proportional Plus Integral (PI) Controller

A PI controller, which is the most used controller function in industry (Haugen TechTeach, 2010), A PI controller is a widely used feedback controller that combines proportional and integral control modes. Proportional control produces an output proportional to the error signal (desired setpoint minus actual process variable), while integral control produces an output proportional to the time integral of the error signal. The combination of these control modes improves the stability and performance of the control system.

According to Sangwan et al., a traditional PI controller accelerates response (Sangwan et al., 2019). However, PI controllers require tuning of the proportional gain and integral time to set their appropriate value based on the thermophysics of the particular heating equipment (Ulpiani et al., 2016). The standard PI controller function is given in this form:

$$u(t) = K_c e(t) + \frac{K_c}{T_i}\int_0^t e\,(t)d\theta \qquad 2.3$$

where K_c is the proportional gain, T_i is the integral time constant and $e(t)$ is the control error. For the PI controller, to avoid the windup generated by the interplay of integral action and saturations, limiting the control law variations is required,

so that the controller output never exceeds the actuator limit u_i^{lim} desired (Hu et al., 2014):

$$u_i = \begin{cases} u_i^{lim} & \text{if } |u_{min}| \geq u_i^{lim} \\ u_{min} & \text{otherwise} \end{cases} \quad 2.4$$

While the PI controller has been used for temperature control in greenhouses, its potential for optimising energy use has not been fully explored in the literature. In this context, some studies have proposed various PI controller tuning methods to improve the closed-loop control performance of greenhouse systems. Here, we summarise and compare four such studies that focus on tuning the PI controller parameters using different optimisation techniques.

The paragraph presents a comparison of four studies that focus on tuning the proportional-integral PI controller parameters to improve the closed-loop control performance of greenhouse systems. The first study by Sigrimis et al. proposes a robust PI controller tuning method for temperature control in greenhouses using first-order plus dead-time models (Sigrimis et al., 1999). The second study by Manonmani et al. optimises the IMC-based PI controller parameters using Genetic algorithm (GA) and Particle swarm optimization (PSO) to achieve a minimum Integral of the error squared (ISE) for set-point change and disturbance rejection (Manonmani et al., 2016). The third study by Amit Sangwan et al. tunes the PI controller using global optimisation algorithms to minimise the weighted sum of the integral of the absolute error (Sangwan et al., 2019). The fourth study by Manuel et al. focuses on designing a PI control strategy for regulating the internal temperature of a greenhouse, compensating for disturbances caused by soil temperature, solar radiation, wind velocity, and outside temperature (Beschi et al., 2014). It is worth noting that while these studies improve the closed-loop control performance of greenhouse systems, the optimisation was not specifically targeted at energy use minimization, so there is still a need for PI controller that minimises energy used.

2.4.3 Proportional-integral-derivative control (PID)

While PI controllers have been used for temperature control in greenhouses, PID controllers have also been studied in the context of optimising greenhouse systems.

Conventional PID controllers, are widely used in industrial and technical applications due to their advantages, such as easy implementation, simple architecture, and exceptional performance. In some applications, as earlier discussed, only PI controllers are considered. Over the past years, the regulation of indoor temperature has received significant attention, and different advanced approaches have been used (Lijun et al., 2018). Notwithstanding, the PID controller stands out as the most commonly used control strategy in both literature and industrial applications. According to Åström and Hägglund, 2000), the PID controller is proposed to be the first solution to be tried when feedback is used on a process, so it is used as part of the chosen controller to verify its energy saving potential based on Chapter 4.

Figure 2.6 (Lautenschlager, 2019a) shows the general structure of controller. The controller part (assumed to be PID) is made up of three parts: the proportional, integral, and derivative parts.

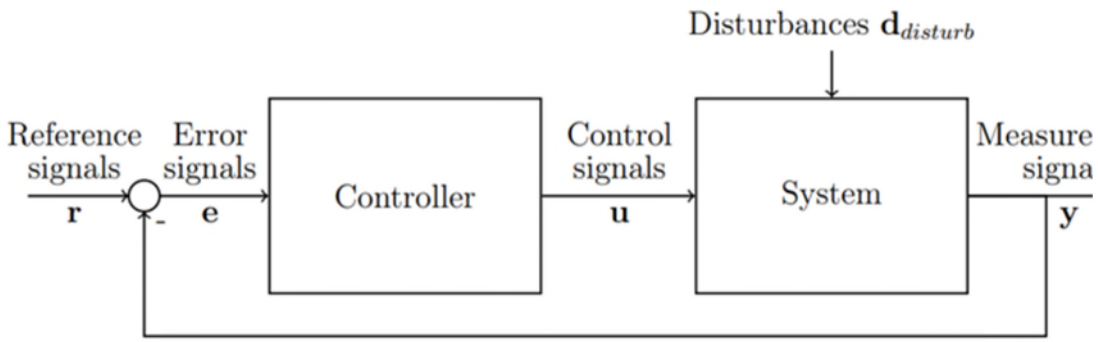


Figure 2.6 scheme of a standard control loop

The controller continuously computes the input or control signal u based on the error signal variable e, which is the difference between the reference signal r and the system's measured output signal y. The controller modifies the control signal

u with the aim of reducing the error signal. [51]. This process is repeated as long as the control system is in effect. In automation equipment, the algorithm is implemented in the microprocessor in the form of a programmed, discrete-time algorithm that computes the control signal u at discrete points in time.

Although, the PID control system is commonly used to maintain greenhouse temperature setpoints and has been used to reduce energy consumption. However, PID control strategies have some drawbacks, such as the tendency to overshoot the desired setpoint; adjusting the controller is a difficult and time-consuming process; and they have difficulties handling external disturbances (Iddio et al., 2020a). The PID controller function is given below:

The control error e is given as:

$$e = y - r \tag{2.5}$$

The control output is given as;

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de}{dt} \tag{2.6}$$

The proportional term K_p in Equation 2.6 is proportional to the current error $e(t)$, the integral term K_i is proportional to the historical accumulation of the error, and the derivative term K_d is proportional to the rate of change of the error. The values of the three terms K_p, K_i, and K_d are typically adjusted by trial and error, which can be a time-consuming process or can be tuned using controller tuning techniques. In addition, tuning the PID controller for optimal performance requires a good understanding of the system's dynamics and external disturbances, which can be challenging in complex systems such as greenhouses. Nevertheless, despite these challenges, the PID control system remains a popular and effective method for controlling greenhouse temperatures and other variables.

The research community prioritises tuning PID controller parameters to achieve better control performance over economic profit (Hu et al., 2014) and tuning to achieve input energy reduction (S. Zhang et al., 2020b), as verified in these works (Bounaama & Draoui, 2011; Horatiu & Andreescu, 2012; Su et al., 2020).

Mahdavian and Wattanapongsakorn used the optimisation method to tune PID controller coefficients for greenhouse lighting control, and they considered the optimisation problem with a varying number of objectives. Their simulation results suggest that increasing the number of separated and independent objective functions can improve the system's output responses (Mahdavian & Wattanapongsakorn, 2014). Haigen Hu et al. proposed a PID controller tuning scheme for a greenhouse climate control system using an evolutionary algorithm based on multiple performance measures. The focus of their work was on tuning the gain parameters of the PID controller to achieve good control performance through step responses such as small overshoot, fast settling time, and less rise time and steady-state error, among other performance measures (Hu et al., 2011). In contrast to other authors, Haigen Hu et al. developed a new methodology for tuning multiple PID controllers in a greenhouse climate control system that incorporates both control performance and economic costs. They used a nondominated sorting genetic algorithm-II to minimise both performance measures and production costs in a simulation experiment. The results showed that the proposed tuning method was effective and could achieve good control performance at a relatively low cost. Although their focus was on energy minimization based on PID control only, there are other works that have considered advanced control methods for optimising greenhouse systems.

The previous discussions focused on various works that used On-off, PI, and PID controllers to optimise greenhouse climate control systems. There are other advanced control methods that can be used to further optimise greenhouse systems. One such method is MPC. MPC is a control method that uses a mathematical model of the system to predict future behaviour and optimise control actions over a finite time horizon. In the context of greenhouse climate control, MPC can be used to optimise multiple inputs, such as heating, cooling, and ventilation, to achieve optimal temperature and humidity conditions while minimising energy consumption.

2.4.4 Model predictive control method

MPC, also known as Receding Horizon Control, is a model-based computer control algorithm (Mogal & Warke, 2013) that uses future predictions to calculate optimal control inputs. To achieve this, MPC uses the process model, the current process state, constraints, setpoint values, and disturbance values identified over a prediction horizon. The optimizer uses the process model to simulate the process over the prediction horizon and can obtain information about the current process state from measurements or a state estimator like the Kalman filter (Haugen, 2021). The constraints specify the maximum or minimum values of process variables, control signals, and state variables, which define the control goal. MPC obtains an optimal control signal or manipulated variable and minimises the objective over the time horizon using the process model (Iddio et al., 2020b). Although the optimisation problem is solved at every time step K, only the first element of the resulting best input trajectory is used in the process (Lautenschlager, 2019b), and the entire calculation is repeated at the next control interval (Mogal & Warke, 2013).

According to Maciejowski, MPC is the most advanced control technique that is superior to PID control as it can handle equipment and safety constraints routinely (Maciejowski, 2002). The history of MPC can be traced back to the early 1960s (Ding et al., 2018), and it has since been widely used in industrial processes. The technique was initially developed as the Dynamic Matrix Control (DMC) method by Cutler and Ramaker at Shell Oil in 1973 (Roberts, 1995). MPC enables the introduction of nonlinear dynamics, constraints, and predictive information, and its linear form is used to solve convex quadratic programming problems online. Recent advances in fast quadratic programming (QPs) solvers have enhanced their applicability in fast systems. Nonlinear MPC (NMPC), on the other hand, is primarily used for slow systems due to its need for significantly more computation (Vukov et al., 2015).

The initial MPC theory integrated DMC and Model Algorithm Control (MAC), which are based on the linear quadratic Gaussian (LQG) and are relatively simple parameter models. Moreover, the Internal Model Control (IMC) was developed

for single input/single output discrete time systems, which are inspired by MAC and DMC, however, these approaches lacked stability. The Generalised Predictive Control (GPC) model was then introduced to address robustness issues, and its theoretical basis is more complete than that of DMC and MAC. However, most industrial processes are highly complex and nonlinear, so adaptive MPC, robust MPC, and NMPC are used. Nevertheless, these models involve long and inefficient computation times. To address this issue, new MPC models like distributed MPC, explicit MPC, and hybrid MPC have been established. Additionally, stochastic MPCs have been developed to handle random production processes (Ding et al., 2018).

MPC has had explosive growth in process industries due to its ability to effectively handle complex systems with difficult control constraints and several inputs and outputs. It has also had an excellent history, with early inclusions in academic literature (Mayne, 2014). Although MPC has been employed in agriculture, it has not yet been fully applied in all field. Presently, MPC is primarily used in agricultural applications such as product processing, machinery, irrigation systems, and greenhouses (Ding et al., 2018).

2.4.5 Model predictive control for a greenhouse system

Accurate modelling of control variables is a crucial component of MPC. MPC for greenhouse systems employs three types of models: first principle, data-driven, and hybrid models. First-principles models are based on heat transfer and mass balance equations and do not require measurement data for parameter identification. However, due to their complexity and the computational time required to solve them, they are infrequently used in MPC.

Data-driven models make use of input and output data to approximate the process behaviour with no explicit physical meaning of the correlation between inputs and outputs. Examples of data-driven models include autoregressive models, neural networks, support vector machines (SVM), FOPDT model and evolutionary regression. Hybrid models, such as state-space models, use measurement data to determine key model parameters and are based on simple

energy balance equations. State-space models are the predominant models used in MPC for greenhouses (Iddio et al., 2020a).

MPC simulations and predictions can be based on any model type that accurately represents the process to be controlled. A flexible model form used in MPC is a discrete-time nonlinear state-space model, represented as follows:

$$x_{k+1} = f(x_{k,}\, u_{k,}\, d_{k,}.) \qquad 2.7$$

$$y_k = g(x_{k,}.) \qquad 2.8$$

x represents the state vector, u is the control or input signal vector, d is the process disturbance vector, and y represents the process variables vector. $f\ and\ g$ are nonlinear or linear vectorial functions (Haugen, 2021). The MPC uses the model to predict the behaviour of the process for future control inputs (Lautenschlager, 2019b). Control input values predicted by continuous simulations over a prediction horizon H_p is calculated using the model of the system with respect to a cost function by the optimizer until an optimal control signal sequence (optimal solution) is found. The prediction horizon, H_p, defines the number of time steps k that the MPC uses to compute the future process behaviour. The MPC optimises the future control input trajectory by minimising a quadratic cost function. The actual measured states, x_k , of the process are fed back to the MPC to initialise the system model for optimisation.

The optimal control sequence also called array (or matrix in multivariable example), u_{opt} is calculated as the optimization problem solution. The predicted control signal and control errors changes are minimized in a least square manner. The first element from the optimal predicted control sequence is then picked and applied to the process as control signal, given as follows:

$$u_k = u_{opt}(1) \qquad 2.9$$

The optimization function to be minimized in MPC is defined as follow:

$$\min_U J \qquad 2.10$$

From equation (2.10) above, J is the objective function, the optimization function to be minimized and its given as:

$$J = \sum_{i=k}^{k+N} (\llbracket e \rrbracket^2 C_e + \llbracket du \rrbracket^2 C_{du}) \quad 2.11$$

J contains norm, which is a measure of the length of a vector. The most common norm used is the quadratic norm, so when the expression $\|.\|_M$ it implies the M-quadratic norm. Where e represents the control error, the matrices C_e and C_{du} are cost or weight matrices that are typically set as constant matrices. and du represents the change in the control signal. So, given a vector $e,$ the expression $\llbracket e \rrbracket^2 C_e$ represents the square of the C_e quadratic norm of e.

u from equation (2.10) is the matrix that contains the control signals r, at each of the point of time of the prediction horizon, given below:

$$u = [u_k, \quad u_k + 1, \ldots., u_k + (N-1), uN] \quad 2.12$$

It indicates the total control signal matrix. A comprehensive description of u matrix can be obtained in (Haugen, 2021). u is then the MPC optimization problem solution. Therefore, from u, u_k is then applied to the system actuator as the control signal given in (2.13) below:

$$u_k = \begin{bmatrix} u_k(1) \\ u_k(2) \\ . \\ . \\ . \\ . \\ u_r(1) \end{bmatrix} \quad 2.13$$

Optimisation formulation is a crucial component of the MPC algorithm, as it allows for the calculation of the optimal control input. Unlike PID, which uses an analytical expression to obtain a control move signal, MPC employs optimisation to either minimise or optimise an objective or cost function subject to particular constraints. When a single objective function is used, it is called univariate

optimisation, while the use of multiple objective functions is referred to as multi-objective optimisation (Iddio et al., 2020b).

Various objective functions have been investigated for agricultural greenhouse applications, such as profit maximisation under specific constraints (Van Ooteghem et al., 2005). El Ghoumari et al. applied model predictive control (MPC) to regulate the temperature in a greenhouse at the Institute for Horticultural and Agricultural Engineering (ITG) at the University of Hannover in Germany. The objective was to achieve temperature control while considering constraints in both manipulated and controlled variables using an online linearization with a low computational burden. They demonstrated several advantages of the MPC algorithm, including improved performance and energy savings, through a real-time experiment using a soft optimal control effort. They compared this MPC scheme with an adaptive PID controller (El Ghoumari et al., 2005).

In another attempt, Ferreira and Ruano proposed a branch-and-bound search algorithm for discrete model-based predictive control of greenhouses with a temperature control strategy based on a mixture of temperature integration and the difference between day and night temperatures. They proposed strategies to achieve faster coverage of the solution search space with a reduced probability of losing the optimal solution. The authors also proposed a simple method to adapt the cost function coefficients online to reduce energy consumption without significantly affecting control accuracy. The methods are briefly described, and a subset of experimental and simulation results are presented (Ferreira & Ruano, 2008).

Qiuying Zou et al. proposed a nonlinear MPC algorithm based on particle swarm optimisation (PSO) to minimise energy consumption while maintaining temperature control in a greenhouse. The proposed controller consists of a predictor, a cost function, and an optimisation technique that uses PSO to solve the nonlinear optimisation problem. The results show that the proposed controller effectively saves energy consumption while maintaining the temperature within the specified range in a plastic solar greenhouse (Zou et al., 2010). Other

attempts, such as the works of (Blasco et al., 2007; W. H. Chen & You, 2020), have considered MPC for minimising energy consumption, with some also considering water consumption in the design.

In summary, this literature review has highlighted various control strategies developed and implemented for greenhouse temperature control. Model predictive control (MPC) is a promising approach that offers improved performance and energy savings, particularly for large-scale greenhouse operations. However, due to its high computational burden and complexity, MPC may not be suitable for small- to medium-scale greenhouse operations. Relatively simple algorithms such as on-off and PI controllers offer cost-effective alternatives that can be useful for small- to medium-scale greenhouse operators. In addition, this review aims to contribute by comparing the energy optimisation performance of on-off controllers to PI, PID, and MPC, as their effectiveness has not been fully verified in the literature. The objective is to identify the most suitable control strategy for greenhouse temperature control that balances energy savings with control accuracy.

2.5 The concept of Internet of things (IoT)

The concept of the Internet of Things (IoT) was first introduced in 1999 by Kevin Ashton at Procter & Gamble (P&G), who recognised Radio-Frequency identification (RFID) as a critical technology for the IoT (Kevin, 2010). The IoT refers to the interconnection of sensors and devices to the internet to enable the exchange of data and messages for smart control and management. Over the past two decades, rapid advancements in communication technologies have led to the widespread use of various short-range and long-range communication protocols, such as RFID, Bluetooth, Zigbee, Bluetooth Low Energy (BLE), LoRaWAN, Narrow Band Internet of Things (NB-IoT)**,** and Wi-Fi, to connect a vast number of devices. The IoT is increasingly recognised as one of the most transformative technologies that could revolutionise the way people interact with things (Y. Wang, 2019).

IoT technology has revolutionised the way information is accessed and exchanged, allowing users to and control various parameters from anywhere in

the world. The agricultural industry has also benefited from this technological advancement, with IoT being used in different cultivation processes such as farm irrigation management (Abioye et al., 2020; Salvi et al., 2017). Traditionally, farmers relied on manual observation-based techniques, which often led to excessive use of pesticides, underuse of water in labour-intensive irrigation, low efficiency in crop production (Patil & Shruti, 2016), and energy waste. However, IoT-automated agriculture, or precision farming, has emerged as a solution to these problems.

Automated agriculture is an intelligent and informational technology that enables farmers to manage land effectively by identifying, analysing, and controlling various farm variables to achieve maximum quality and profitability with little or no human intervention. Scholars and farmers have made significant efforts to combat these issues, resulting in the birth of this new technology (Georgieva et al., 2016; Sreekantha, 2016). As technology acceptance grows, the implementation of IoT in agriculture is expected to increase and lead to even more efficient and productive crop yields.

In the context of greenhouse management, IoT can be used to monitor and control various parameters such as temperature, humidity, CO2 levels, and lighting conditions, as well as irrigation and fertilisation systems. IoT-based greenhouse systems can also be integrated with cloud computing, data analytics, and machine learning algorithms to optimise resource utilisation, reduce operational costs, and improve crop yields. Overall, the application of IoT in greenhouses has enormous potential to transform the agriculture sector by enabling precision farming, reducing waste, optimization of resources, and improving sustainability. In the next sub-section, the elements that constitute the IoT concept will be discussed.

2.6 The IoT elements

The building blocks that constitute the IoT concept is described below. the understanding of these elements will help readers to gain a better insight into the function and meaning of the IoT. The six main elements that are needed to deliver

the functionality of IoT in any application are illustrated in Figure 2.7 below (Al-Fuqaha et al., 2015).

For the identification section of the elements, usually, when things communicate in IoT, they need to authenticate or verify each other, this requirement is essential to know that they talk to the intended party.

Figure 2.7 The IoT elements

Typically, authentication involves identification (Alpár et al., 2016). There are several methods of identification for the IoT technologies, examples are; Near Field communication (NFC)**,** RFID, Optical readable bar code, identification allows objects or things to be linked to the information related with the particular object and, this information can be retrieved from a server, if the mediator is connected to the network (Er. Pooja Yadav & Er. Ankur Mittal, 2016), equally, identification of real-world things is important for the IoT to match and name services with their demand. other examples are ubiquitous codes (uCode) and electronic product codes (EPC)**.** In addition, addressing the different IoT objects or things is essential to make a distinction between things ID and its address. Things ID such as “Temp1” refers to the things name, for example a particular temperature sensor and thing’s address inversely denotes the things address within a communication network Al-Fuqaha et al., 2015). The assigning of unique identifier enables objects of different types to be part of the internet (Bkheet & Agbinya, 2021).

The IoT sensing element involves gathering environmental data from connected objects in a network and transferring it to a database, data warehouse, or cloud. The collected data is then analysed to perform a specific task, depending on the

application requirement. Devices used in sensing include actuators, smart sensors, and wearable sensors. Different IoT products are usually realised with Single Board Computers (SBCs) such as Arduino Yun, BeagleBone Black, Raspberry Pi, etc., which are integrated with sensors and have built-in Transmission Control Protocol/Internet Protocol (TCP/IP) with security functionalities. These devices are connected to a central management portal (Al-Fuqaha et al., 2015), such as IoT Cloud, ThingSpeak, etc., to log the amassed data in an application.

Wireless Sensor Network (WSN) technologies are used in the sensing stage of the IoT. Research on WSNs dates back to the 1980s with the Distributed Sensor Networks (DNS) programme created to aid communication in military operations at the Defence Advanced Research Projects Agency (DARPA). WSN technology has since spread as a suitable, adaptive, and cost-effective technology for monitoring, automation, safety, and control in various fields (Villaverde et al., 2012; Q. Wang & Balasingham, 2010). This promising technique has evolved over the years, with farmers currently using it in modern-day agriculture for the improvement of crop productivity, site-specific management, and the enhancement of traditional farming methods (Al-turjman, 2019; Rodríguez et al., 2017; Ruiz-Garcia et al., 2009). WSN technology senses and amasses environmental data in farmland, greenhouses, and other complicated environments where human access is difficult, impractical, and dangerous (Ferentinos et al., 2016; Messai & Seba, 2016). It aids in the control of farming processes and provides real-time information to users for better decision-making (Khairunnniza-Bejo et al., 2018).

WSNs are evolving as a potential tool for achieving the global goal of cost-effective automated agriculture with the advent of Micro-Electro-Mechanical systems (MEMs) technology through the creation of small and cheap sensors with sensing capabilities that constitute several small nodes that are ubiquitous in operation and self-organised in nature (Jan et al., 2017; Ojha et al., 2015). WSNs are collections of a few to many battery- or renewable energy source-powered autonomous sensor nodes or motes (Balamurali & Kathiravan, 2015;

Buratti et al., 2009; Messai & Seba, 2016; Varghese & Sharma, 2019; J. Zhang & Varadharajan, 2010) spatially spread in a given area that cooperatively sense, compute, and communicate environmental parameters in hops to a processing centre, also known as a sink node or base station (Watt et al., 2019), for local control. In order to reduce the cost of WSNs and for energy constraints, only one node, which is the sink node, is generally allowed to transmit data from the wireless sensor network to the internet or outside world through a gateway using long-range connections for user remote access (Barrenetxea et al., 2008).

WSN is an evolving technology that has gained the attention of manufacturing industries, researchers, technical manufacturers, and enterprising farmers over time. These groups have embraced this technology in different agricultural processes (Khairunnniza-Bejo et al., 2018) ranging from pre-seeding soil analysis (Khairunnniza-Bejo et al., 2018; Lvova & Nadporozhskaya, 2017), plant management (Ferentinos et al., 2016; Thakur et al., 2018), smart irrigation (Goap et al., 2018; Lakhwani et al., 2019), crop harvesting (Bapat et al., 2017), and post-harvest management (Aung & Chang, 2014; Lokke et al., 2011; J. Wang et al., 2015). A WSN system comprises intelligent nodes that collect application-oriented data. In this context, the sensor node networks perform three major functions: sensing, computing, and communicating data using a combination of hardware, software, and algorithms (Aqeel-Ur-Rehman et al., 2014). The parameters measured include but are not limited to surrounding temperature, humidity, soil moisture, etc. (Thakur et al., 2019).

Communication is a vital component of the IoT, enabling WSN devices to connect and share data with one another over a network. IoT communication technologies link heterogeneous objects to provide specific smart services, such as file sharing, messaging, and data transmission (Burhan et al., 2018). There are various IoT communication protocols available, including wireless fidelity (Wi-Fi), Bluetooth, LTE-Advanced, RFID (Al-Fuqaha et al., 2015), Zigbee, Near Field Communication (NFC), Z-Wave, LoRa, Cellular, Sigfox, and Neul.

To better understand the different networking protocols available, researchers have classified them into three categories: short-range protocols (such as

Bluetooth, Zigbee, and NFC), medium-range protocols (including Z-Wave and Wi-Fi), and long-range protocols (such as LoRa, Cellular, Sigfox, and Neul) (Elhadi et al., 2020). However, in another work, Wi-Fi was classified as short-range (Y. Wang, 2019),. The most suitable protocol for an agricultural application should be determined based on its suitability for the specific application rather than solely on its category or popularity. As Xu et al. (Xu et al., 2022) noted, selecting the right protocol for an agricultural IoT application is crucial and should never be done randomly. Therefore, careful consideration of the strengths and weaknesses of each protocol, as well as the specific requirements of the application, is essential. Figure 2.8 below illustrates the commonly used data transmission protocols in agricultural IoT.

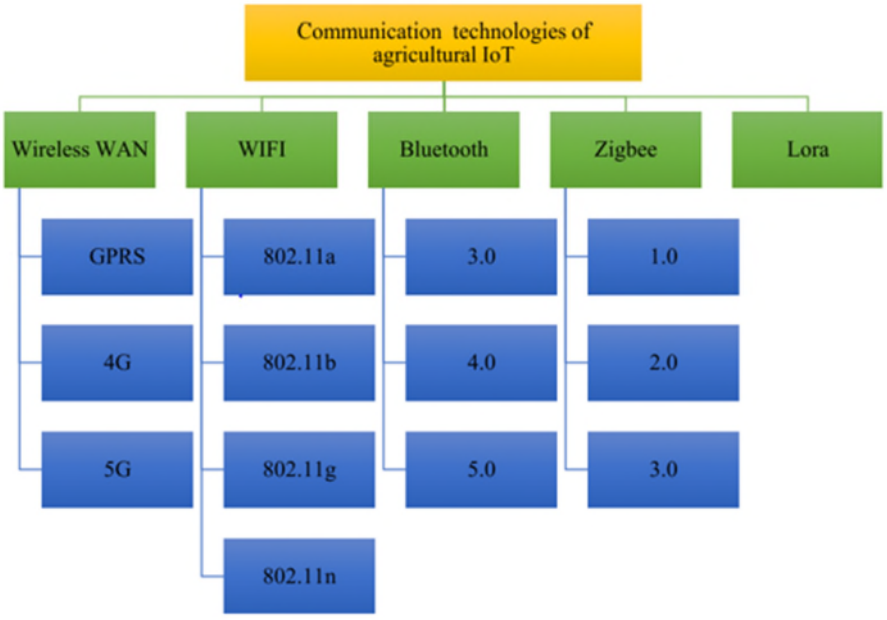


Figure 2.8 communication technology commonly used in agricultural IoT

The Figure 2.8 shows the commonly used data transmission protocols in agricultural IoT applications.

Computation plays a crucial role in processing the data collected from various objects or environments in an IoT application through the use of sensors (Burhan et al., 2018). This processed data is then used for decision-making purposes (Bkheet & Agbinya, 2021). The computational ability of the IoT is dependent on

different hardware platforms, including microprocessors, microcontrollers, field-programmable gate array (FPGAs), System-on-a-chip (SOC)**,** and software applications, which serve as the "brain" of the IoT. Popular hardware platforms for running IoT applications include Arduino, Intel Galileo, Raspberry PI, Beagle Bone, Cubieboard, WiSense, and T-Mote Sky. Additionally, lightweight operating systems such as TinyOS (Levis et al., 2005), Riot OS (Baccelli et al., 2014), and LiteOS (Cao et al., 2008) are used to deliver IoT functionalities, specifically designed for IoT environments (Al-Fuqaha et al., 2015).

In the realm of IoT, services can be broadly categorised into four classes: identity-related Services, Information Aggregation Services, Collaborative-Aware Services, and Ubiquitous Services. Identity-related services are crucial for other services since every piece of software or application that connects real-world objects to the virtual world must first identify those objects. IoT information aggregation services collect and summarise raw sensory measurements to provide a comprehensive view of the environment. Collaborative-aware services leverage the aggregated data to make informed decisions and take actions accordingly. Ubiquitous services aim to provide Collaborative-aware services whenever and wherever they are needed. These categories have been identified by researchers in the field (Al-Fuqaha et al., 2015) and provide a framework for understanding the range of services that IoT applications can offer. By leveraging the unique capabilities of IoT devices, such as sensors and connectivity, these services can improve efficiency, reduce costs, and enhance the overall user experience.

Semantic in IoT refers to the process of extracting knowledge from the data collected by various devices and sensors, enabling them to provide intelligent services. This involves tasks such as resource discovery and utilization, as well as information modelling, which helps to identify relationships between different data points. Semantic processing enables machines to detect and assess data to determine the best course of action for delivering a specific service. In essence, semantic processing acts as the brain of IoT, directing requests to the appropriate

resource. This requirement is supported by Semantic Web technologies such as the Resource Description Framework (RDF). (Al-Fuqaha et al., 2015).

2.6.1 Structure and the design of an IoT based control system

An IoT-based greenhouse control system typically comprises three sub-systems or layers. The first layer is the monitoring sub-system, which is responsible for environmental perception or awareness by collecting and acquiring data on various environmental parameters using sensors. The second layer is the information processing subsystem, where the collected data is processed and analysed by a server. This layer enables the visualisation, storage, and analysis of the acquired data. Moreover, in many cases, the information processing subsystem is designed to make decisions based on the monitored data. Finally, the communication layer facilitates the exchange of data between the first two layers. This layer handles the transmission of data, signal processing, and the sending of commands to the actuator for system modification (H. Li et al., 2021).

To effectively control environmental changes in a greenhouse, it is essential to have a precise monitoring system. The greenhouse monitoring sub-system serves as an environmental awareness or perception subsystem. Its purpose is to monitor and measure environmental, soil, and crop parameters and then send the acquired data to the next subsystem. Traditionally, greenhouse monitoring systems are based on wired communication. Although wired systems guarantee data transmission security, this type of monitoring technique is inflexible and complicated to meet the necessary requirements for long periods involved in cultivated fields and periodic changes in cultivation types (Bai et al., 2019).

WSN offer several advantages over wired systems, such as low cost, low power consumption (Yang et al., 2017), large scale (S. Chen et al., 2015), convenient installation (Peng & Han, 2016), small size, intelligence (Yick et al., 2008), flexibility, extensibility, and deployment (Bai et al., 2019). The more detailed function of the three subsystems in IoT based control system design is discussed in the next section.

2.6.2 IoT Control Systems Hardware Resources

The optimisation of greenhouse energy through a control system involves both hardware resources and control algorithms or strategies. The hardware components include sensors, actuators, and controllers. The use of mini and microcomputers as control systems has become increasingly popular over time, from their application in tasks such as climate control, data logging, and alarming (Udink ten Cate, 1984) to the present microcontrollers. Thermostats for heating control were introduced in the late 1950s, and computers have been used in greenhouse automation since then (Vijverberg & Strijbosch, 1968). Sensors, such as those used for temperature and humidity control, are essential for feedback- and feedforward-based control schemes. However, it has been shown that horticultural sensors are prone to substantial errors, which can lead to increased energy consumption (Bontsema et al., 2011). In spite of that, the effective use of the control system is nevertheless capable of coordinating hardware resources to minimise energy consumption in the greenhouse.

Sensing technologies play a crucial role in monitoring changes in the environment, allowing for the safe and efficient use of natural resources, and promoting green technology. In agricultural applications, sensors are used to monitor environmental parameters such as moisture and temperature to ensure optimal crop growth. Agricultural sensors can be classified based on a few factors, including the environmental property to be monitored and the power supply requirement. Active sensors, which require an external power supply, radiate some energy to trigger a response from the object of interest, while passive sensors capture incident radiation emitted from objects (Dargie & Poellabauer, 2011).

Agricultural sensors can be broadly divided into three categories based on their properties: chemical sensors, physical sensors, and mechanical sensors (Abdullah & Barnawi, 2012). Chemical sensors measure chemical properties such as pH and dissolved oxygen; physical sensors measure physical properties such as temperature and pressure; and mechanical sensors measure mechanical properties such as strain and vibration. The selection of sensors are

dependent on various factors, including the environmental property to be monitored (Pajares et al., 2013; Sreekantha, 2016). Two types of sensors are commonly used in greenhouses: biochemical sensors, which monitor data related to the plants, soil, crops, or other microorganisms and are used to detect the presence of chemical elements or pesticides; and physical sensors, which are used to monitor environmental variables such as the air temperature, humidity, irradiation, and pressure data (Bersani et al., 2020). Depending on the application, a node or microcontroller can have one or more sensor types integrated (Capable & Processor, 2010; Sinclair, 2001).

Once the sensor has detected and measured the physical, chemical, or biological stimuli, it converts the stimuli into an analogue or digital signal (Das et al., 2011). The signal is then passed through signal conditioning stages, including an amplifier or an attenuator to change the signal power to the required range and a filter to translate the measured signal into a usable signal through a filtration process. The signal is then passed through an analogue-to-digital converter (ADC), which digitalizes the signal and makes it ready for transmission, storage, and remote visualisation purposes through a processing unit such as a microprocessor, FPGA, etc. Actuators are devices that convert an electrical signal into a non-electrical signal (De Marcellis & Ferri, 2011), and they are used to complete the loop for automatic control. Electrical actuators such as valves, fans, and air coolant plants convert electrical signals from sensors into physical actions. In unfavourable winter climate conditions, greenhouses use various active or passive heating systems or actuators; thermal models designed for greenhouse application in such regions will account for the heating or heat storage materials (Sethi et al., 2013). Cooling components, such as fog systems, pad and fan evaporative cooling, shading, and ventilation, have been used in hot regions to maintain an indoor climate favourable for crop growth (Abdel-Ghany & Kozai, 2006). In an agricultural scenario, for example, actuators could be used to control irrigation systems, lighting, or ventilation (Akyildiz et al., 2002; Dargie & Poellabauer, 2011; Kalaivani et al., 2011; Q. Wang & Balasingham, 2010).

While IoT hardware resources have been extensively used in various greenhouse applications, such as hydroponics, intelligent lighting control systems, greenhouse irrigation systems, and crop-sensitive data monitoring, research on reducing total heating energy demand using IoT-based control systems is limited in the literature. Only a few studies have explored this topic (Sagheer et al., 2021) and Šaš & Pejić (Šaš & Pejić, 2021), but the IoT-based automatic control systems in these studies did not use greenhouse energy models, making it challenging to determine their impact on reducing energy consumption. Moreover, proprietary control systems for indoor agriculture are not affordable for small- to medium-sized greenhouse operators.

Knowledge Gap

Three significant contributions were made based on the literature review conducted in this study.

- Firstly, the existing physics-based dynamic air temperature model was optimised to simulate the thermal process of a greenhouse equipped with IoT hardware resources for indoor temperature prediction. The model was simplified through optimisation techniques to fit the indoor and outdoor data collected from the prototype system, which is crucial for effective controller design. This contribution addresses a gap in the literature, where only a few studies have explored the minimization of energy demand using IoT, and in cases where IoT solutions are used, models are not employed to implement IoT-based automatic control system solutions.
- Secondly, the suitability of four optimised control strategies, including the On-off temperature control approach, which is a simple and low-cost temperature control method, was evaluated for minimising energy use compared to PI, PID, and MPC control algorithms. This contribution aims to address the need for a simple, inexpensive control algorithm that everyday farmers can use to optimise greenhouse energy use.
- Lastly, a unique IoT-based automatic control system design and implementation approach was proposed that provides cost-effective solutions for small to medium-sized greenhouse or indoor growth facility

producers. This contribution addresses the need for economical options for small to medium-sized greenhouse or indoor growth facility producers despite the availability of various proprietary indoor agricultural control systems.

This literature review has established the main findings and gaps in the research related to greenhouse temperature control using IoT-based control systems. These findings and gaps will provide a foundation for the subsequent chapters of this research. In Chapter 3, the greenhouse models used in this study will be described. Chapter 4 will compare the energy optimisation performance of different control strategies, including on-off, PI, PID, and MPC controllers, to identify the most suitable control strategy for minimising energy use while maintaining accurate temperature control. In Chapter 5, the data acquisition system used in this study will be discussed. The remaining chapters will provide a comprehensive understanding of the experimental setup, data collection, and analysis procedures used in this research to achieve the stated objectives.

CHAPTER 3 Temperature Model

To effectively control the heating energy in greenhouses, a good understanding of the greenhouse environment or climate system is necessary. This chapter focuses on the comparison of a physics-based temperature model with a data-driven FOPDT model. The physics-based model is developed using energy balance equations to construct the differential equation of an energy model. This approach, also known as "first principles“, modelling, is essential to describing the climate inside the greenhouse and the functionality of the control equipment. A dynamic energy simulation model helps to simulate the temperature dynamics and facilitates understanding of the complex mechanisms involved in the thermal process of greenhouse operation. The model-based control design is necessary to understand the behaviour of the system and to make predictions for control purposes (Speetjens et al., 2010).

3.1 Design of the physics-based model

Figure 3.1 depicts a schematic of the greenhouse climate interacting with the control inputs, the outside environment, and the greenhouse crop. The schematic summarises the typical fluxes that affect the greenhouse environment. The air temperature, T_{air}, is affected by fluxes Q_{vent}, Q_{sun}, Q_{trans}, and Q_{heat_ex}, which are replaced by Q_{heater} in this work. To minimise energy input, Q_{vent} and Q_{heater} terms were grouped to be the control input, Q_{energy} (Van Beveren et al., 2013).

The greenhouse climate modelled in this work assumes a "perfectly stirred tank" approach, where the air is treated as one homogeneous entity and spatial variability is ignored in the model (Roy et al., 2002). A representative value of indoor air temperature, Tin (°C), is used to represent the state of the greenhouse (Katzin et al., 2022a). The dynamic changes in the climate of the greenhouse are realised through the energy and mass flows obtained from the differences in the energy and mass content between the indoor and outdoor air and the control energy inputs, as shown in Figure 3.1 (Van Beveren et al., 2013):

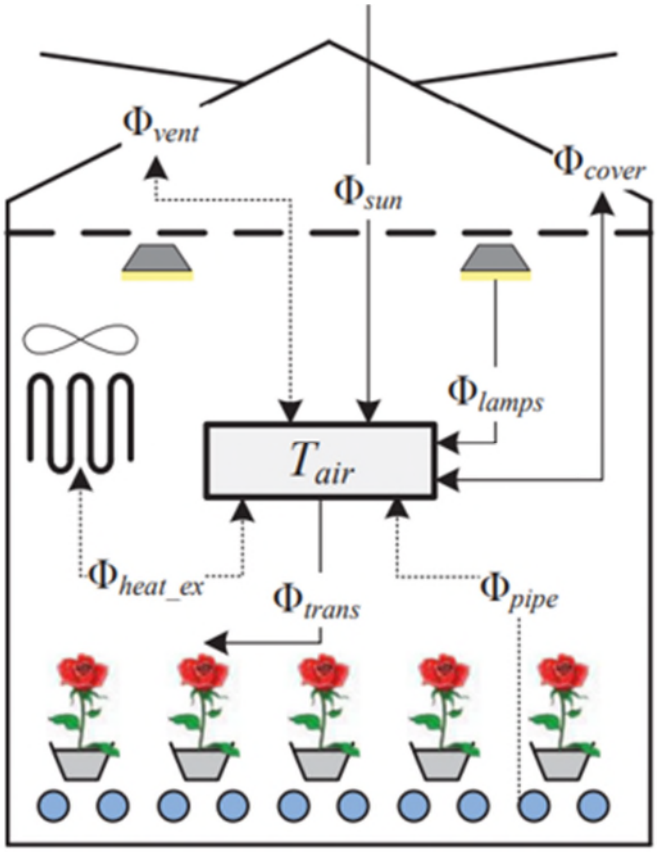


Figure 3.1 Greenhouse climate system

The control inputs for the greenhouse temperature model are the heating system temperature $u(W)$ and the roof vent position u_v. In addition, three exogenous inputs or disturbances are considered: solar irradiation v_1 (W/m^2), the outside temperature v_2 (℃) and v_3 wind speed m/s.

The dynamic model is represented in state space form as:

$$\dot{x} = f(\mathrm{x}, \mathrm{u}, \mathrm{v}, \mathrm{p}, \mathrm{t}) \quad \text{with } x(\mathrm{t}_0) = \mathrm{x}_0 \qquad 3.1$$

where t is the time, x is the state, u is the control input, v is the external input or disturbance, and p denotes system parameters.

To build the physics-based temperature model, the greenhouse climate models described by Van Beveren et al. (Van Beveren et al., 2013) and Van Henten (Henten, 1994) were used as a basis, and modifications were made to account for the additional actuators used in the prototype greenhouse. In particular, the

model of Van Beveren et al. was significantly modified, and the work of Liang et al. (Liang et al., 2018) was incorporated. The function f in equation (3.1) describes the greenhouse air temperature T_{in} over time, which is given by:

$$\frac{dT_{in}}{dt} = \frac{1}{Ccap}(Q_{sun} - Q_{cov} - Q_{trans} + Q_{lamp} + Q_{energy}) \quad 3.2$$

For simulation purposes, Q_{energy} is defined as follows:

$$Q_{energy} = -Q_{vent} + Q_{heater1} + Q_{heater2} + Q_{heater3} + Q_{heater4} \quad 3.3$$

In this work, equation 3.3 was modified to include the three additional heaters used in the greenhouse lower layer to maintain the desired microclimate temperature of 23℃ suitable for the tomato plants. Q_{trans} in Van Beveren et al. was replaced with Q_{soil}. Note that Q_{energy} represents the energy input, and a negative energy flux indicates that Q_{energy} extracts heat from the greenhouse air (cooling), while a positive energy flux adds heat to the greenhouse air (heating).

The Q_{energy} term in equation 3.3 is where the introduced additional heaters are incorporated, thus deviating from the model of Van Beveren et al. In their model, Q_{vent} represents the heat exchange with outdoor air due to natural ventilation. However, to modify and adapt this model for the built prototype greenhouse and to follow the approach proposed by Katzin et al. (Katzin et al., 2022b), the model was modified as shown in equation 3.3. Specifically, three additional heaters (lamps) and primary control system 3 were included, represented by $Q_{heater3}$ (heating or cooling), as the main heating system in the upper layer (rack) of the greenhouse to maintain the environmental temperature.

Control systems 1, 2, and 4, represented by $Q_{heater1}$, $Q_{heater2}$, and $Q_{heater4}$, respectively, were designated to maintain the required microclimate temperature for maximum plant growth, as shown in Figure 3.2. The total energy input, $Q_{heater3}$, was used as the control input u. The external inputs or disturbances, represented by v_t in equation 3.4:

$$v_t, t_o, \leq t \leq t_f \quad 3.4$$

Figure 3.2 The greenhouse system window

where measured climate data including solar radiation, outdoor temperature, and wind speed were obtained from the Cranfield University climate weather station as shown in figure 3.3.

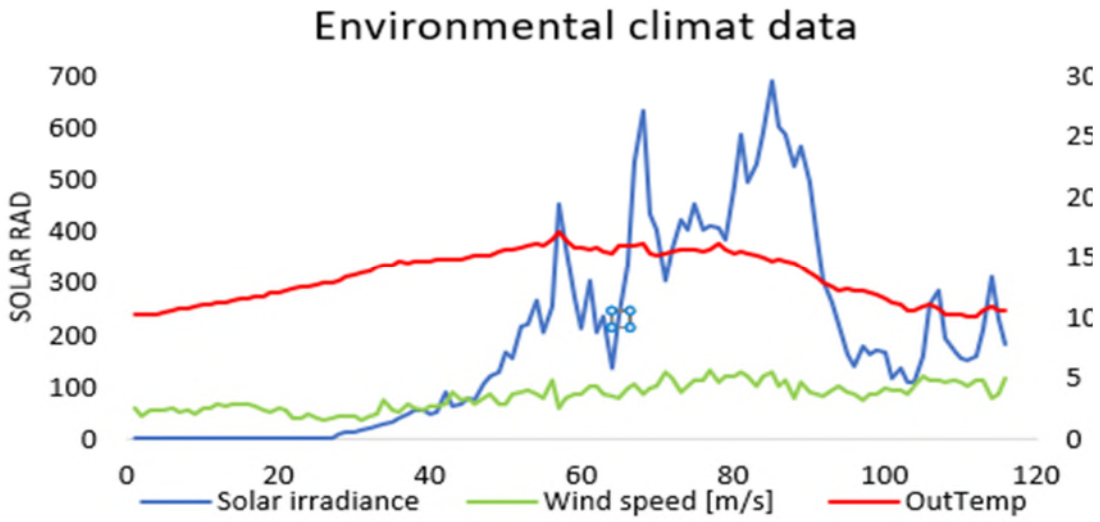


Figure 3.3 Climate weather data

Although the indoor data that was used for the validation was from the prototype from this greenhouse system, the exogenous data were obtained from the Cranfield weather station as shown in Figure 3.3.

3.2 Model equations

This section provides mathematical models for the energy and mass flows that affect the greenhouse air temperature, T_{in}. Equation 3.2 provides the overall model for T_{in}, and Equation 3.5 models the specific heat capacity of the greenhouse air, $Ccap$ [J/m^2 ℃]. The $Ccap$ is adopted from Liang et al.'s work (Liang et al., 2018) and, given by:

$$Ccap = \rho_a C_a V_g \qquad 3.5$$

where ρ_a is the air density $\sim 1.29\, kg.m^{-3}$ under standard conditions, C_a is the specific heat capacity of air at constant pressure $1000\, J.kg^{-1.}℃^{-1}$, and V_g is the greenhouse volume m^3.

Q_{sun} represents the heat load imposed on the greenhouse by the sun and is computed using the solar radiation model derived by Henten (Henten, 1994) and simplified by Liang et al., as given below:

$$Q_{sun} = C_{rad} Q_{rad} A_g \ (W) \quad 3.6$$

Where C_{rad} accounts for transmission of the roof, interception of transmitted solar radiation by structural components of the greenhouse, and conversion of absorbed solar energy by the canopy into a sensible heat load on greenhouse air. Q_{rad} is external solar radiation, expressed per square meter of soil $(W\ m^{-2})$, and A_g is the greenhouse surface area in m^2.

Equation 3.7 (Liang et al., 2018) describes the convective heat loss through the cover:

$$Q_{cov} = Q_{coe}(T_{out} - T_{in}) A_c \ (w\ m^{-2}) \quad 3.7$$

Here, Q_{coe} is the heat transfer coefficient of the cover material, given in $(W\ m^{-2}k^{-1})$, T_{out} and T_{in} are the outdoor and indoor temperature, respectively, (℃) and A_c is the greenhouse cover material area in (m^2).

Q_{trans} in Equation 3.2 is replaced with Q_{soil} in equation 3.8. The Q_{trans} in equation 3.2 is the energy released to the greenhouse air through crop transpiration, the Q_{soil} model used in Liang et al. for the computation of heat transfer between internal air and soil Q_{soil} was used to replace it, given below as:

$$Q_{soil} = K_{coe}(T_s - T_{in})\ A_g \quad 3.8$$

Here, K_{coe} is the soil heat transfer coefficient, and T_s is the soil surface temperature, set to be 20°C.

For the simulation purposes, natural ventilation is achieved through the opening of the roof window (as shown in Figure 3.2), and Q_{vent} is the heat lost through natural ventilation. Equation 3.9 models Q_{vent}:

$$Q_{vent} = \rho_a C_a N_{ven}(T_{out} - T_{in})\ A_{ven} \quad 3.9$$

Here, ρ_a is the air density $\sim 1.29\ kg.m^{-3}$ under standard conditions; C_a is the specific heat of air at constant pressure $1000\ J.kg^{-1.}{}^{\circ}C^{-1}$; N_{ven} is the natural ventilation rate $m\ s^{-1}$ given in equation 3.10 below. N_{ven} is the natural ventilation

rate $m\,s^{-1}$, which is a function of the window opening and external wind speed and is used as a control variable in the model. Equation 3.10 models N_{ven}:

$$N_{ven} = \kappa + \theta w + vwu_w \ (m\,s^{-1}) \qquad 3.10$$

Here, w is the external wind speed $m\,s^{-1}$, κ, θ, and v are the ventilation rate parameters related to air flow, air expansion, and air infiltration respectively and u_w is the window opening, and its value is set to 0 (off) or 1 (on).

Lastly, from equation 3.9 above, A_{ven} is the greenhouse effective ventilation area, the model of (Boulard & Baille, n.d.) was used in this part, however the modified version of it by (Liang et al., 2018) was used in the computation as given below:

$$A_{ven} = 22A_w \sin(\alpha/2) \quad (m^2) \qquad 3.11$$

Here, A_w is the roof window total area, m^2; and α denotes the window opening angle, °. The next four sets of equations were used to calculate the heat imparted to the greenhouse air by the four heaters used in this experiment (Liang et al., 2018):

$$Q_{heater3} = U_{h3}K_{coe3}(T_{p3} - T_s) \quad (W) \qquad 3.12$$

Here, $Q_{heater3}$, is the energy supplied from the primary (main) heater to the greenhouse air; U_{h3} is the heater opening, it value is set either 0 (off) or 1 (on); K_{coe3} is the heat transfer coefficient $W\ ℃^{-1}$ and T_{p3} is the heater3 temperature, given a value of 255.

$$Q_{heater1} = U_{h1}K_{coe1}(T_{p1} - T_s) \quad (W) \qquad 3.13$$

In equation 3.13, $Q_{heater1}$, is used to calculate the energy supplied from the Heater1 to the greenhouse air; U_{h1} is the Heater 1. Its value could be set to either 0 (off) or 1 (on), K_{coe1} is the heat transfer coefficient $W\ ℃^{-1}$ and T_{p1} is the Heater1 temperature, set to a value of 50.

$$Q_{heater2} = U_{h2}K_{coe2}(T_{p2} - T_s) \quad (W) \qquad 3.14$$

Where $Q_{heater2}$, is the energy supplied from the Heater2 to the greenhouse air; U_{h2} is the Heater2 opening, it value is set 0 (off) or 1 (on); K_{coe2} is the heat

transfer coefficient W °C^{-1} and T_{p2} is the Heater2 temperature, set to a value of 25.

$$Q_{heater4} = U_{h4} K_{coe4} (T_{p4} - T_s) \quad (W) \tag{3.15}$$

Here, $Q_{heater4}$, is used to calculate the energy supplied by Heater4 to the greenhouse air; U_{h4} is the heater4 control, it is set to either 0 (off) or 1 (on); K_{coe4} is the heat transfer coefficient W °C^{-1} and T_{p4} is the Heater 4 temperature, is equally set to a value of 50.

The artificial lighting installed in the greenhouse equally added heat to the greenhouse air. The heat was computed by using the equation below:

$$Q_{lamp} = \eta \, Ep \frac{F_{on}}{100} \quad (W \, m^{-2}) \tag{3.16}$$

Here, η is the portion of the lamp's electric energy consumption that is converted into heat and released into the greenhouse air, Ep is the rated electric power of artificial lighting that we installed $(w \, m^{-2})$ and F_{on} is the percentage of the lamp that is on (%).

From the previous discussion in section 3.1, the simulation algorithm for the greenhouse air temperature is derived from equation 3.2, which is given again below having the Q_{soil} as the substitute of Q_{trans}:

$$\frac{dT_{in}}{dt} = \frac{1}{Ccap} (Q_{sun} - Q_{cov} - Q_{soil} + Q_{lamp} + Q_{energy}) \tag{3.17}$$

To convert this equation into a state space model, equation 3.18 is used, which is expressed as:

$$T'_{in} = \frac{1}{Ccap} (Q_{sun} - Q_{cov} - Q_{soil} + Q_{lamp} + Q_{energy}) \tag{3.18}$$

Next, the model equations discussed in section 3.2 were substituted, which gives equation 3.19 for simulation:

$$T'_{in} = \left(\frac{1}{Ccap}\right) * \left[\left(C_{rad} * Q_{rad} * A_g\right) - \left(Q_{coe} * (T_{out} - T_{in}) * A_c\right) + \left(K_{coe} * (T_s - T_{in}) * A_g\right) + +Q_{lamp} + (-Q_{vent} + Q_{heater1} + Q_{heater2} + Q_{heater3} + Q_{heater4})\right] \quad 3.19$$

The simulation algorithm for this model can then be implemented in the steps below as follows: First, the state variable was initialised before beginning the simulation loop.

Initialization: T_{in}_k = T_{in}_initial

Inside the simulation loop:

- T_{in}_k is limited between T_{in}_min = 0℃ and T_{in}_max = 23℃
- The input signals Q_{energy}_k, T_{out}_k, and Q_{rad}_k was set before storing T_{in}_k in an array for plotting.
- To compute the time derivative, the equation below was used:

$$T'_{in} = \left(\frac{1}{Ccap}\right) * \left[\left(C_{rad} * Q_{rad} * A_g\right) - \left(Q_{coe} * (T_{out} - T_{in}) * A_c\right) + \left(K_{coe} * (T_s - T_{in}) * A_g\right) + +Q_{lamp} + (-Q_{vent} + Q_{heater1} + Q_{heater2} + Q_{heater3} + Q_{heater4})\right]$$

- Finally, the prediction was computed using the Euler forward method: $T_{in\_}kp1$ = T_{in}_k + dt * $dT_{in}\_dt\_k$
- Time index shift T_{in}_k = $T_{in\_}kp1$
- and the simulation loop was repeated until it reaches the desired end time.

3.2.1 Simulation and validation

Using the developed model, simulation was conducted to analyse the dynamic behaviour of the system in response to various inputs, such as disturbances and changes in outdoor temperature. Climate data in 8-minute intervals from the Cranfield University greenhouse process control system room was collected and used for the simulation. To validate the greenhouse air temperature, the measured indoor temperature data from the IoT system built was used. The

model was calibrated by obtaining some parameter values from Liang et al.'s work (Liang et al., 2018) and using greenhouse physical properties to determine others, as shown in Table 3.1 below. These parameter values were chosen because their model has been validated and confirmed to show promising results. Moreover, the volume of the greenhouse used in their experiment for calibration was almost the same as that used in the prototype greenhouse system.

Table 3.1 Parameter values adopted from literature.

Parameters	value
κ	0.1
θ	1.0857
v	0.4444

In the simulation, a few assumptions were made, including;

- that the greenhouse air is homogeneous,
- the floor area is planted directly instead of using pots,
- and the feedback influences of crop growth on the air temperature are ignored.

The fourth-order Runge-Kutta algorithm in Matlab was used to solve the system equation 3.19 above, and the Matlab code is provided in Appendix 3.1. Figure 3.4 shows the simulation output, which displays the greenhouse's initial setup temperature of 22.9℃ ≈23℃, which corresponds to the indoor temperature measured by the temperature sensor from the IoT based control system shown in Figure 3.5.

When different values of T_{out} and Q_{rad} values were used, the value of the indoor temperature indicates that the T_{in} is directly influenced by the outdoor temperature, solar radiation, and the heater, as the corresponding values changed in proportionally to the varying outdoor weather condition.

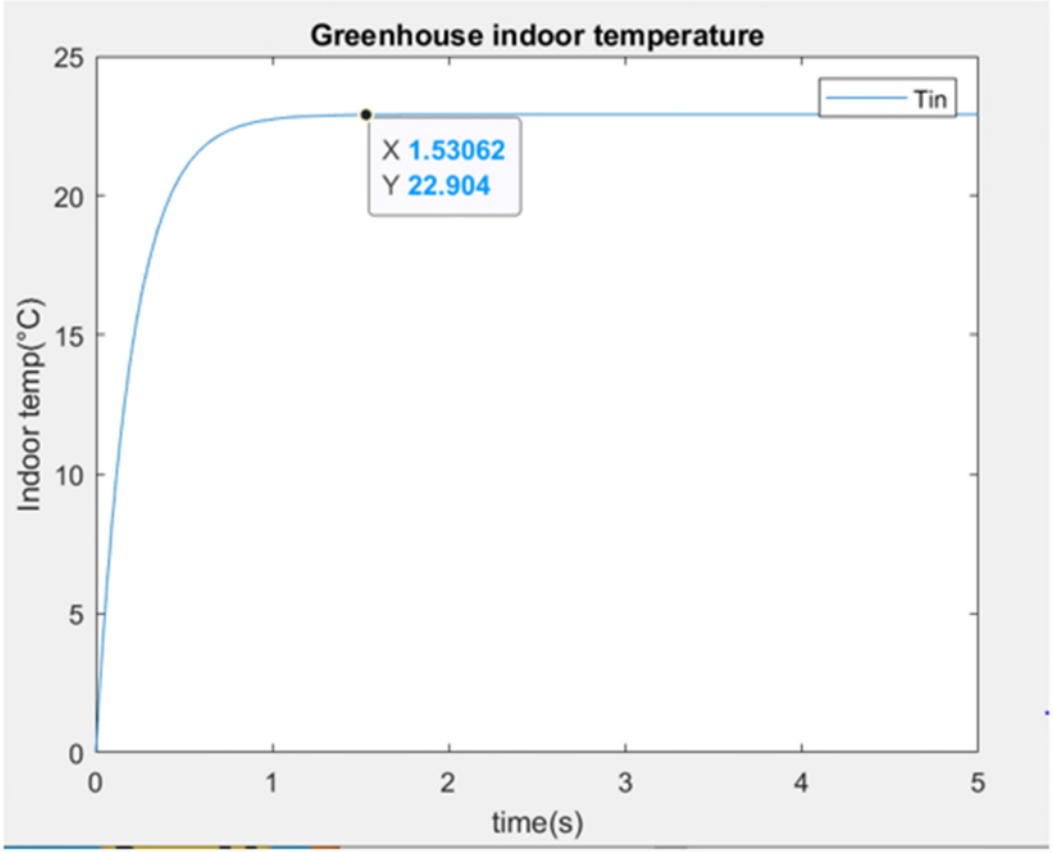


Figure 3.4 Simulation output of the non-linear system

The measured data from the simulation in Figure 3.4 above was validated using the Figure 3.5 below, which is the measured indoor (Tin) data from the built IoT based control system.

3.2.2 Validation result

The simulation results showed a close resemblance between the obtained simulated output and the measured values, as seen in Figure 3.4 and Figure 3.5 respectively. The initial steady state value of the greenhouse temperature from the simulation was 22.90 °C, which is only 0.6 °C lower than the measured value of 23.5 °C in Figure 3.4. While this difference may be considered acceptable, it should be noted that the temperature difference between the actual and simulated greenhouse air varied over time, resulting in a mean difference of 0.5 °C and a variance of 2.62. These results suggest that the modified model is reasonably effective in simulating the climate behaviour of the greenhouse,

although there may be some limitations in accurately predicting temperature variations over time.

	Tin	Heating
10:30:32.75Z	23.5	255
10:30:34.833Z	23.75	202
10:30:35.893Z	23.25	0
10:30:38.003Z	23.75	43
10:30:39.063Z	23.25	42
10:30:40.143Z	23.5	0
10:30:41.233Z	24	43
10:30:45.393Z	24.25	42
10:30:47.503Z	24	128
10:30:51.754Z	23.75	84
10:30:52.773Z	24.5	214
10:30:53.823Z	24.75	255
10:30:54.963Z	24.5	0
10:30:55.943Z	24.75	43

Figure 3.5 Measured input and output data from IoT based control system

This observation suggests that the greenhouse system's dynamics changed with time. The changes in dynamics could be attributed to several factors, including changes in outdoor temperature, solar radiation, and greenhouse air temperature. Furthermore, the physics-based model was simplified as a SISO system for controller design using an optimisation approach; this is discussed in Section 3.3 below. The model was fit using the greenhouse data by optimising two parameters.

3.3 Design of the simplified physics-based model

The simplified physics-based model for greenhouse temperature control is derived from the original differential equation for air temperature, which considers the heat transfer between the internal and external environments of the greenhouse. The model is expressed in equation 3.17, which is referred to again below as:

$$\frac{dT_{in}}{dt} = \frac{1}{Ccap}\left(Q_{sun} - Q_{cov} - Q_{soil} + Q_{lamp} + Q_{energy}\right) \quad 3.17$$

The model shows the change in greenhouse air temperature with respect to time is proportional to the difference between the heat gains and losses. To simplify the model and make it amenable for controller design, equation (2) below is used:

$$\frac{dTin}{dt} = \frac{1}{Ccap}(Q_{sun} - Q_{cov} - Q_{soil} + Q_{energy}) \quad 3.20$$

Here Q_{energy} is replaced by:

$$Q_{energy} = (-Q_{vent} + Q_{heater}) \quad 3.21$$

Which is the energy term provided by heating i.e., heat gains from the power source with the difference between the heat losses due to ventilation. The model parameters are determined based on the greenhouse air properties, including its heat capacity, convective heat loss through the cover, and heat transfer between the air and soil as previously discussed, however, they are restated here for emphasis as given:

$Ccap$ represents the heat capacity of the greenhouse air given as:

$$Ccap = \rho_a C_a V_g \quad 3.5$$

Q_{sun}, represents the heat load imposed on the greenhouse by the sun, given below:

$$Q_{sun} = C_{rad} Q_{rad} A_g \ (W) \quad 3.6$$

The convective heat loss via the cover Q_{cov} is described below as,

$$Q_{cov} = Q_{coe}(T_{out} - T_{in})A_c \ (w\, m^{-2}) \quad 3.7$$

The heat transfer between the internal air and soil is described as:

$$Q_{soil} = K_{coe}(T_s - T_{in})\, A_g \quad 3.8$$

Lastly, for simulation purposes, in other to simply the model to a simpler version, the previous vent models in equations 3.9 - 3.11 were replaced with the vent model described in the work of Van Beveren et al. (Van Beveren et al., 2013), given below as:

$$Q_{vent} = Q_{ven} * R_{hoa} * C_a * (T_{out} - T_{in}) \quad 3.22$$

Equation 3.22 shows that the heat lost through natural ventilation is proportional to the temperature difference between the indoor and outdoor environments, the ventilation area, and the overall heat transfer coefficient.

3.3.1 Simplified model simulation and validation

To simulate the system's behaviour, a numerical algorithm is implemented in Python that solves equation 3.20 (The code is in Appendix 3.2) based on the input parameters and initial conditions. To validate the model, the simulation results are compared with the measured data from the greenhouse, the same way as described in Section 3.2. Some assumptions are made in the simulation, including the homogeneity of the greenhouse air and the assumption that the floor area is directly planted instead of using pots. The feedback effects of crop growth on air temperature are also ignored in the model.

The simulation is performed in this section to validate the improved model in equation 3.20 using the IoT-based control system's (closed loop) measured indoor temperature data depicted in Figure 3.6. Validation entails determining if the improved model adequately represents the pilot greenhouse. In the following section, a method for validating the applicability of the proposed simplified model is described.

The work done in this section aims to estimate two model parameters in Python that can predict indoor temperature based on equation 3.6 This is achieved by optimising the predicted output to match the experimental measurements of the real system shown in Figure 3.6 below.

Using an optimisation technique, the following steps were followed to formulate the parameter estimation problem as an optimisation problem. The least squares method was used, and the objective function to be minimised is the sum of squared prediction errors given below.

$$SSE = \sum_{k=1}^{N} e(k)^2 \quad 3.23$$

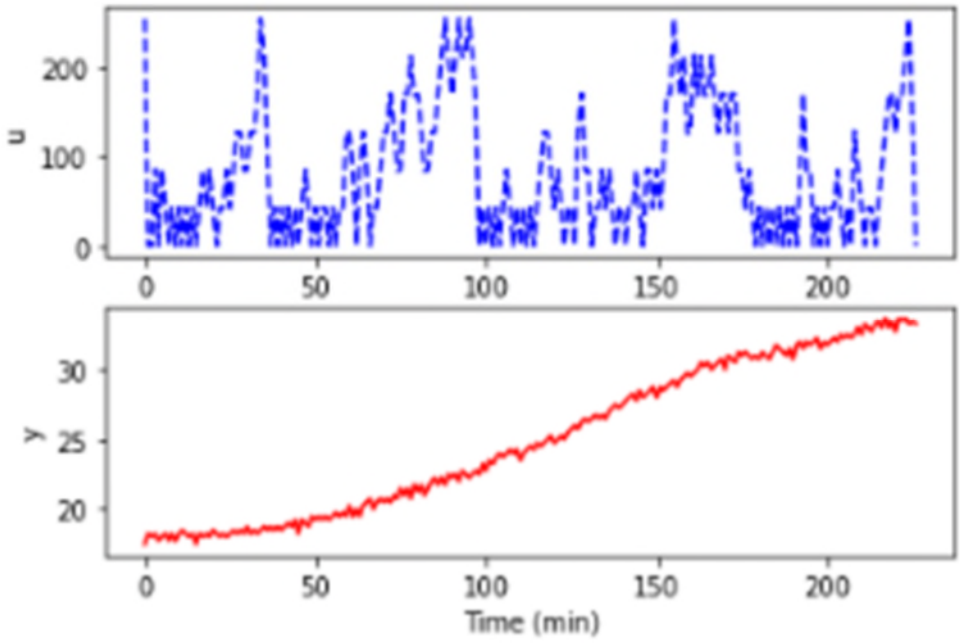


Figure 3.6 (a, b) IoT-based control system measured input and output data

where $e(k)$ represents the prediction error, which is the difference between the observed temperature $Tin_{obs}(k)$ and the predicted indoor temperature (Tin_{pred}), given as:

$$e(k) = Tin_{obs}(k) - Tin_{pred}(k) \quad 3.24$$

where Tin_{obs} is observed temperature values which are taken from Figure 3.6 and Tin_{pred} is the predicted indoor temperature that was calculated through simulations. The goal is to optimize the prediction errors and make the predicted output as similar as possible to the experimental measurements. The optimization problem was solved using the Python library SciPy, which provides a function called “curve_fit” that implements the least squares method. The “curve_fit” function requires as input the model function equation 3.20, the experimental data (indoor temperature measurements), and an initial guess for

the parameters. The output of the “curve_fit” function is the estimated parameters that optimize the objective function (i.e., the sum of squared prediction errors).

The estimated model parameters were used to simulate the Indoor temperature of the greenhouse over a period of one day (using the 25 May 2022 data). The simulation results were compared to the experimental measurements to assess the accuracy of the model. The comparison showed that the model with the estimated parameters was able to predict the indoor temperature of the greenhouse with reasonable accuracy. This confirmed the validity of the physics-based dynamic model and the effectiveness of the parameter estimation technique used in this section.

Overall, the work in this section demonstrates the importance of parameter estimation in model-based control of indoor temperature in greenhouses. The optimization technique used in this section can be applied to other models and experimental data to estimate model parameters and improve the accuracy of the model predictions. The simulation condition is described below.

The model in Equation 3.20 was solved numerically in Python; a static model of the environmental data of the solar radiation [W/m^2], the wind speed [m/s], and outdoor temperature [°C] measured on 25 May 2022 shown in Figure 3.3 was used as inputs to the model.

The known thermophysical properties of the greenhouse cover, soil, vent, and heater parameters are used as input parameters to the model. Values shown below in Table 3.2 are the other required parameter values that were obtained from the literature.

Table 3.2 Parameter values adopted from literature

Parameters	value
$Crad$	0.1
C_a	1000
$Rhoa$	1.29

To predict the air temperature accurately, two unknown model parameters, which are the total ventilation flux from indoor to outdoor air Q_{ven} in equation 3.22 and the soil surface temperature, T_s, in equation 3.8 are estimated through optimization technique as previously described in python, and the values are shown in Table 3.3.

Table 3.3 Optimized model parameters

Estimated Parameters	Identified Value
Q_{ven}	0.29
T_s	11.05

Through the least square method, the parameters were obtained; in this method the unknown parameters are adjusted until the difference between the measured and the simulated temperature is minimized in the least square sense. This method gives good estimates after several iterations with an initial guess ($T_s = 20.58$ and $Q_{ven} = -5.88$). The evaluated model from the experimental data with a minimum sum of squared errors (SSE) is realized.

3.3.2 Validation results

Figure 3.7 shows the results of the simulation for the validation of the model the time interval 0.0–20.0 min, the simulated indoor air temperature based on the estimated model is plotted together with the real measured indoor temperature.

```
Initial SSE Objective: 39807.692303073345
Final SSE Objective: 91.1413391989764
Ts: 11.05362082839811
Qven: 0.2949219722610455
```

(a)

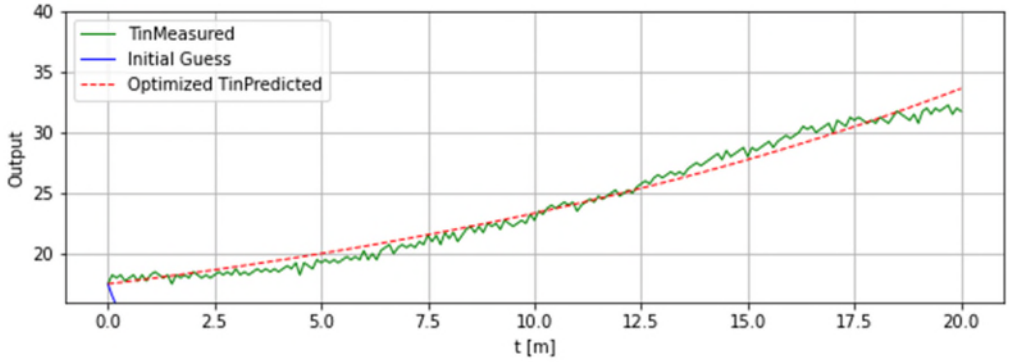


Figure 3.7 (a) Optimisation result; (b) measured versus simulated temperature [°C]

The simulation runs with the initial state value set equal to the real measured state at t = 0.0. The highest discrepancy is found within 12.5–17.5 min between the simulated indoor air temperature and the actual measured temperature during this period, which is roughly 2°C. The modified physics-based model can correctly estimate the inside temperature. The initial SSE objective in the values is Figure 3.7 (a) above and the final SSE objective after optimization is depicted. The optimization-based model adaptation strategy for simulating greenhouse air temperature offers promising results: it produces a model that fits well and yields predictions that are approximately equal to the values of the measured air temperature data. To evaluate the performance of the physics-based model, it is compared to a FOPDT model that was fitted using system identification techniques. This comparison will determine which model better predicts the greenhouse temperature and will be used to test for energy minimization between the controllers. The procedure for this comparison is explained in the following sections:

3.4 Design of the first order plus dead time (FOPDT) model

Physical nonlinear models are mathematically complex (Montoya-Ríos et al., 2020), in this study, the FOPDT model is employed to estimate the dynamic response of the greenhouse temperature for control design. The FOPDT model is a well-known technique in process control that can represent the temperature

response to changes in heating input power using a simple linear model. The model identification technique is utilised to fit the parameters of the FOPDT model, which include gain, time constant, and dead-time or time delay. These parameters serve as initial tuning constants for the controller. System identification is then performed using input-output data gathered from experiments conducted in the experimental greenhouse with the built IoT control system.

The captured data are used to build a linear model of the greenhouse system using the FOPDT structure, which captures the effect of the main environmental conditions and heating system on the indoor air temperature. The data was collected approximately every 40 second and stored on the Arduino IoT Cloud platform. The resulting controller model can be used to design PID form controllers or other types of controllers (Burn & Cox, 2020) such as MPC (Choomkasien et al., 2017). The comparison of the FOPDT model with the simplified physics-based model will be used to determine the model that better predicts the greenhouse temperature for controller design and energy minimization between the controllers.

3.4.1 FOPDT model simulation and validation

The FOPDT model was fitted to the experimental data discussed in the previous section using the curve_fit function from the scipy.optimize package in Python. The FOPDT transfer function was defined using equation 3.25 (code in Appendix 3.3), given as:

$$y = k_p(1 - np.\exp(t - theta_p)/tau_p) \qquad 3.25$$

where the process gain, k_p, process time constant, tau_p, and process delay, $theta_p$, were used. The specific data file was imported, and the curve_fit function was applied with initial guesses of k_p = 1.0, tau_p, = 2.0, and $theta_p$= 4.0 to estimate the FOPDT model parameters. After extracting the optimised parameters, the FOPDT response was simulated using the np.linspace function to generate a time vector and the FOPDT transfer function with the optimised parameters.

The simulated output was then interpolated onto the same time points as the measured output using the np.interp library function. The experimental and simulated data were plotted using the matplotlib.pyplot.plot function, and the root mean squared error (RMSE) was computed to assess the accuracy of the FOPDT model. The difference between the experimental temperature data and the simulated temperature data was computed using the numpy library, and the square root of the mean of the squared differences was also computed. The units used for RMSE were $(temperature\ unit)^2$, which is temperature units squared.

Lastly, the scipy.optimize library was used to minimise the difference between the steady-state input energy required for a temperature of 23°C and the input energy estimated by the FOPDT model. The minimise function was used to minimise the absolute difference between the steady-state input energy required and the input energy estimated by the FOPDT model. The optimal input energy required was extracted using the x attribute of the result of the minimise function.

It is worth noting that only the energy from the primary heating system, $heater_3$, was used as input energy data for the FOPDT model estimation. The influence of solar radiation and outdoor temperature were excluded from the estimation because their inclusion resulted in FOPDT model parameters that were not able to accurately represent the measured data.

3.4.2 Validation results

Figure 3.8 shows the results of the simulation for the validation of the FOPDT model in Equation 3.20. In the time interval 0.0–20.0 min, the simulated indoor air temperature based on the estimated model is plotted together with the real measured indoor temperature.

```
Estimated model parameters:
Kp = 66891.42106688074
tau_p = 112223.697690609
theta_p = 1.107294120048038
RMSE = 0.5733440876219992
```

(a)

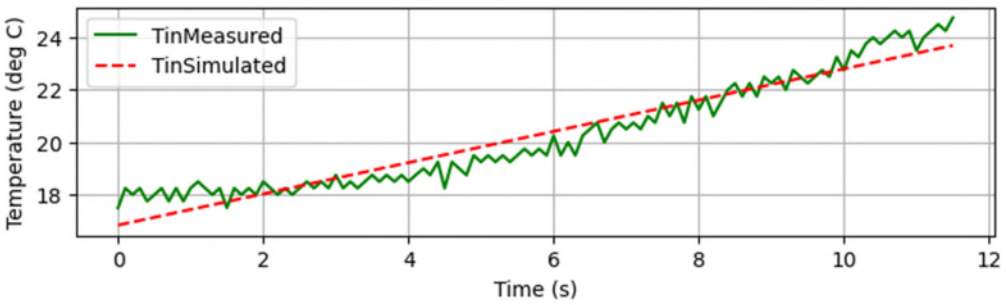


Figure 3.8 (a) Optimisation result; (b) measured versus simulated temperature [°C].

As shown in Table 3.4 (a) and (b) below, the comparison between the prediction of the physics-based model and the FOPDT model is to decide which was best at predicting the indoor temperature of the prototype greenhouse.

Table 3.4 performance matrices results.

Physic-based model	FOPDT model
Estimated Physics-based model parameters: Final SSE Objective: 91.1237730808773 Initial SSE Objective: 39807.692303073345 RMSE = 0.6733142690304332 mean_absolute_error: 0.5607873493858413 R-squared: 0.9805519897267015	Estimated FOPDT model parameters: Kp = 66891.42106688074 tau_p = 112223.697690609 theta_p = 1.107294120048038 RMSE = 0.5733440876219992 mean_absolute_error: 0.4796071551406652 R-squared: 0.923749965153443
(**a**)	(**b**)

To achieve this, three matrices were used: First, the mean squared error (MSE) values were measured. The MSE is a measure of the average squared difference between the predicted values and the actual values. Next, the mean absolute error (MAE) values, the MAE is a measure of the average absolute difference between the predicted values and the actual values. Finally, the R-squared is a measure of the proportion of the variance in the target variable that is explained by the model.

The objective of this analysis is to determine the best model for predicting indoor temperatures in a greenhouse. Two models were considered: a physics-based model and a first order plus dead-time (FOPDT) model. The models were

evaluated based on their ability to predict indoor temperatures using the temperature data measured in the prototype greenhouse. The performance of the models was evaluated using three metrics: MSE, MAE, and R-squared (R2) value. The physics-based model had an MSE of 0.673, a MAE of 0.561, and an R2 value of 0.981. On the other hand, the FOPDT model had an MSE of 0.573, a MAE of 0.480, and an R2 value of 0.924.

Based on these metrics, the physics-based model outperforms the FOPDT model in predicting indoor temperatures. The physics-based model had a lower MSE and MAE and a higher R2 value than the FOPDT model. The lower MSE and MAE indicate that the physics-based model's predictions are closer to the actual values than the FOPDT model's predictions. The higher R2 value indicates that the physics-based model better captures the variation in the data. Therefore, the physics-based model is chosen as the best model for predicting indoor temperatures. This model is based on the principles of heat transfer and considers the greenhouse’s physical characteristics, which makes it a more accurate predictor of indoor temperature than the FOPDT model.

In conclusion, the implication of the finding is that, since the analysis shows that the physics-based model is a better predictor of indoor temperature compared to the FOPDT model. The physics-based model's superior performance in terms of the evaluation metrics, along with its physical basis, make it the most suitable model for use in designing the greenhouse controllers.

CHAPTER 4 Controller Design

The control system is a crucial component of any automation system as it regulates physical processes to achieve desired performance criteria. In the context of greenhouse energy optimisation, controllers are essential for regulating greenhouse temperatures to ensure optimal plant growth conditions and minimise energy consumption.

In Chapter 3, the physics-based model was shown to provide better predictions of indoor temperatures. Thus, this chapter focuses on implementing two sets of control algorithms based on the physics-based model to maintain a desired setpoint temperature of 23°C. The first set of control algorithms, an On-off and an IMC-tuned PI, were used in simulation to regulate the system to the grower's desired temperature. Their performances were compared based on metrics such as overshoot and error minimization. In the second set of simulations, four additional control algorithms, including On-Off, PI, PID, and an MPC, were implemented to evaluate their energy minimization capability. Instead of a fixed setpoint temperature, varying temperatures were tested to assess the performance of each controller. The results were evaluated and discussed in detail in this chapter.

Overall, this chapter provides an overview of the implementation procedure of PI and ON-OFF controllers using the physics-based temperature model and examines the performance of the other four controllers for energy minimization in the context of greenhouse energy optimisation.

4.1 Simulation of static operating point

It was assumed that the grower desired to maintain 23 °C that is required for the plant to grow in the greenhouse. To accurately calculate the required power needed to maintain the desired 23℃, a static response of $T_{in}{}^{0}$ from the greenhouse model was calculated. The required static operating point ($Q_{energy}{}^{0}$, $T_{in}{}^{0}$), was calculated using the simplified model in equation 3.20. The required static control input energy ($Q_{energy}{}^{0}$) needed to bring the indoor temperature to a constant value of 23 °C was determined by calculating the operating point from

the model and setting the obtained air temperature dynamics equation to zero in Python. The simulation was intended to test whether the model could be used in varying weather conditions, as shown in Figure 4.1 below. Three simulations were done using one day's data, with input disturbances (outdoor temperature and solar irradiation) adjusted to 10.5 °C and 108.2 W/m^2, 16.0 °C and 333.0 W/m^2 and finally 22.0 °C and 400.0 W/m^2, respectively, for 6000 s. (Code is in Appendix 4.1) Figures 4.1 a–c below show the needed input energy to maintain the reference indoor temperature value under different circumstances. It was found that the desired indoor temperature could be maintained even if the outdoor conditions changed, provided there is proper actuator (either a heating or cooling system). Results showed that the model could be used effectively to maintain indoor temperatures regardless of outdoor conditions.

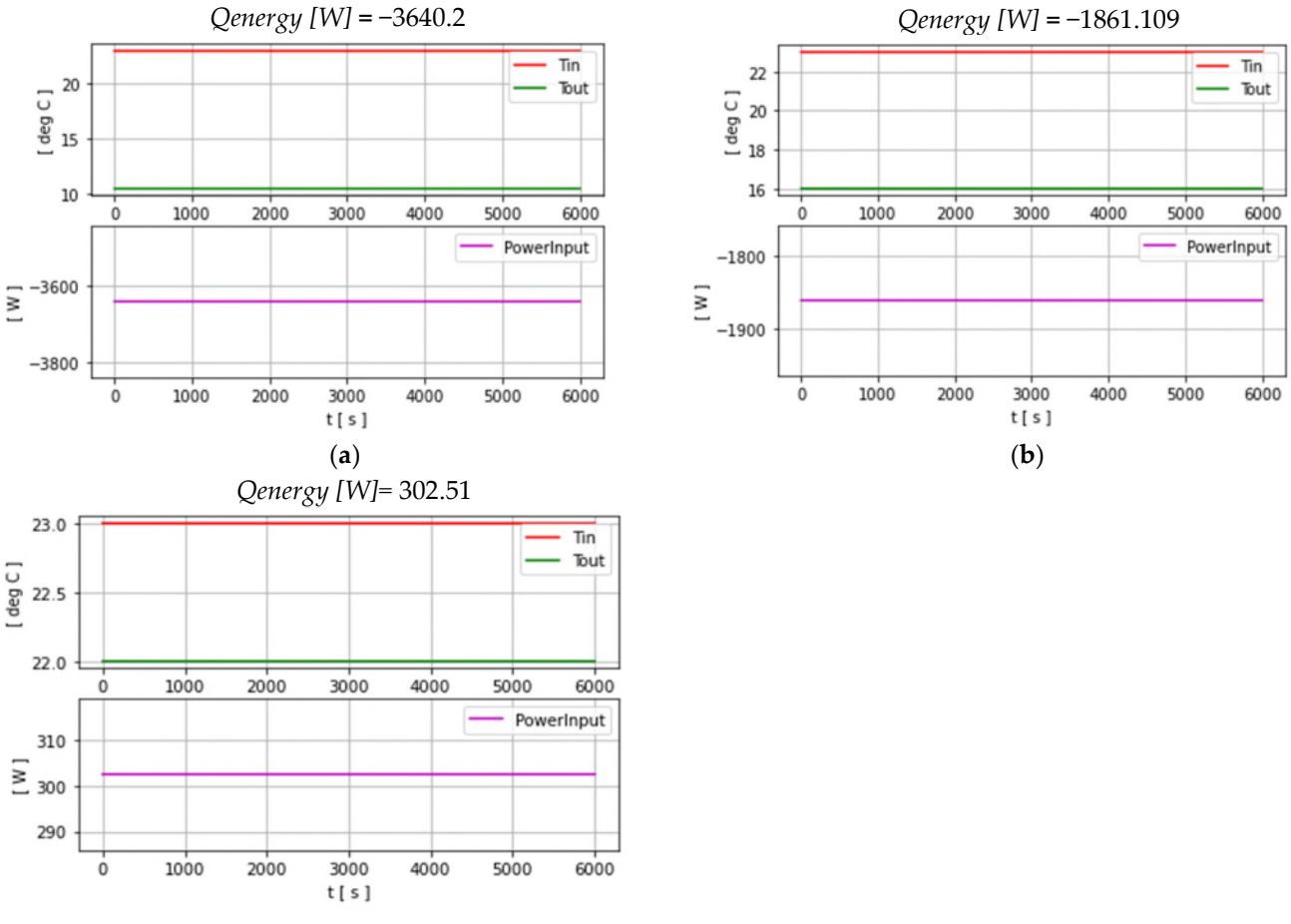


Figure 4.1 a–c Simulation result

4.2 On-off and PI controller design

The dynamic behavior of a process defines how the measured process variable changes over time because of the controller input and any disturbance factors; understanding process dynamics is critical when designing and tuning a controller. The controller input in this experiment is a pulse-width modulated (PWM) signal from the Arduino microcontroller in the designed IoT based control system for this study, this part is described in Chapter 6.

The signal shown previously in Figure 3.6(a) that the IoT-based control system's measured input data varies between 0 and 255, and it was used to control the state of a relay that operates the actuator (heating system). Depending on the value of the PWM signal, the relay varies to switch on or off the heater, thereby affecting the process variable. However, due to external or environmental disturbance factors shown previously in Figure 3.3, which highly influenced the measured air temperature, it is essential to calculate the required static input energy to enforce the indoor temperature to the desired static (steady state) temperature value of 23 °C using the system model.

To design a PI control system, it is necessary to estimate the dynamic properties of the process. For this requirement, the calculated steady-state operating point in Figure 4.1a was used in the simulation of a Doublet test to confirm how the indoor air temperature responds to changes in the input signal and disturbances, as shown in Figure 4.2. Generally, in a Doublet test, a system is allowed to stabilize at some step input signal and corresponding temperature. The equilibrium is then disrupted either by raising or lowering the step input signal, and the resulting dynamic response time is measured (Cook, 2020).

The dynamic process data shown in Figure 4.2 (Code is in APPENDIX 4.2) below was used for the design of the control system. The key properties of the process dynamics were estimated by fitting a first-order plus dead-time (FOPDT) model to the dynamic process response data. The FOPDT model parameters obtained were used later in subsequent Section to design the tuned controller and are shown in Table 4.1.

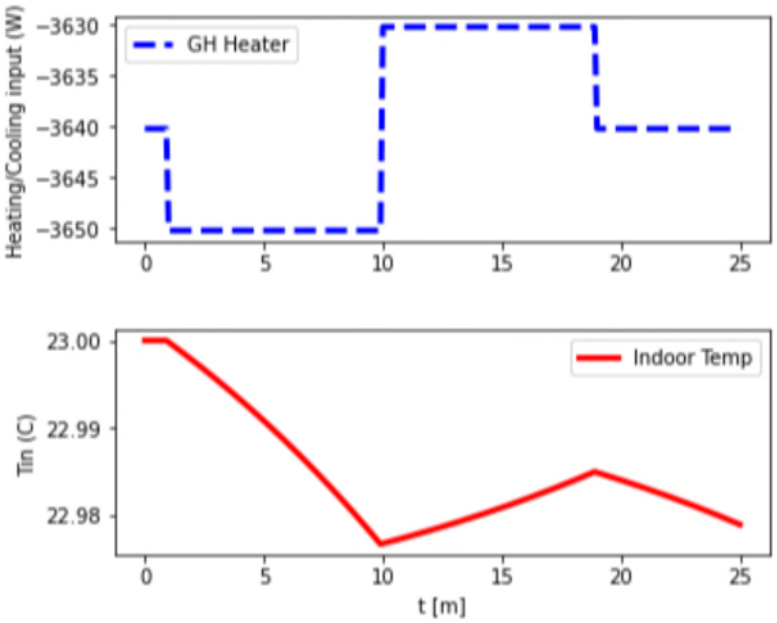


Figure 4.2 Simulated process response to Doublet test

Although two controllers were designed, the results of three controllers were evaluated in the simulation, resulting in a total of three controllers described in this section: an On-off control, a PI controller with gains determined by trial and error, and a tuned PI controller design. A brief discussion of the two-control method is provided below, along with an evaluation of their performance.

Table 4.1 FOPDT process parameters.

Parameters	Values
K_p	0.04
τ_p	151.9
θ_p	11.8

An On-off controller is a simple control based on two rules, and it may be used as a substitute for a PID controller, particularly in temperature control. The control signal input value varies between a maximum of 3640.2 when in heating mode and −3640.2 (W) minimum when in cooling mode, using the calculated from the

static test as the required energy input to maintain 23^0C. The control function was given in equation 2.2 – 2.3 and repeated below as:

$$u_{(t)} = \begin{cases} u_{max} & if\ e\ \geq 0 \\ u_{min} & if\ e < 0 \end{cases} \quad 2.2$$

Where;

$$e\ = r(t) - y(t) \quad 2.3$$

$\mathrm{e}(\mathrm{t})$ is the control error, $r(t)$ is the setpoint or reference signal and $y(t)$ is the process output measurement, u_{max} is the maximum control signal, usually 100%, and u_{min} is usually 0%. However, u_{max} is set as $3640.2 \text{ and } u_{min}$ is used as cooling in the simulation, it was set as −3640.2, The output of the control algorithm is the power supply set as the Q_{energy} given in equation (4.19) for the simulation of the control system.

$$Q_{energy} = (-\ Q_{vent} + Q_{heater}) \quad 3.21$$

The value is given below as;

$$Q_{energy} = Qenergy_{haeting} = 3640.2 \quad or\ Qenergy_{cooling} = -3640.2 \quad 4.1$$

A PI controllers require tuning of the proportional gain and integral time to set their appropriate value based on the thermophysics of the heating equipment as discussed previously. in this section the trial-and-error method was first used to obtain the first PI control output and later the estimated process constants from the FOPDT model fitting were used in a correlation to obtain initial estimates of the second PI control tuning parameters. The standard PI controller function is given previously in equation 2.4 is restated here:

$$u(t) = K_c e(t) + \frac{K_c}{T_i}\int_0^t e\,(t)d\theta \quad 2.4$$

where K_c is the proportional gain, T_i is the integral time constant and $e(t)$ is the control error. For the PI controller, to avoid the windup generated by the interplay

of integral action and saturations, the control law variations are set to a limit, so that the controller output never exceeds the actuator limit u_i^{lim} restated as follows:

$$u_i = \begin{cases} u_i^{lim} & \text{if } |u_{min}| \geq u_i^{lim} \\ u_{min} & \text{otherwise} \end{cases} \quad 2.5$$

4.2.1 Simulation result and discussion

The controller design simulation was executed in Python for 1440 min each; these were intended to create a 1-day test from 6 am to 6 am of the next day. For the trial-and-error controller tuning, the parameters were randomly selected until these; K_c set to 80 (W/K) and the T_i value was set to 5.9 (s) were obtained. Although the values were obtained through the trial-and-error method, however, these controller parameters can be changed to different values. The trial-and-error PI controller's parameters are shown in Table 4.2 below.

Table 4.2 PI tuning trial and error values.

Parameters	Values
K_c	80
T_i	5.9

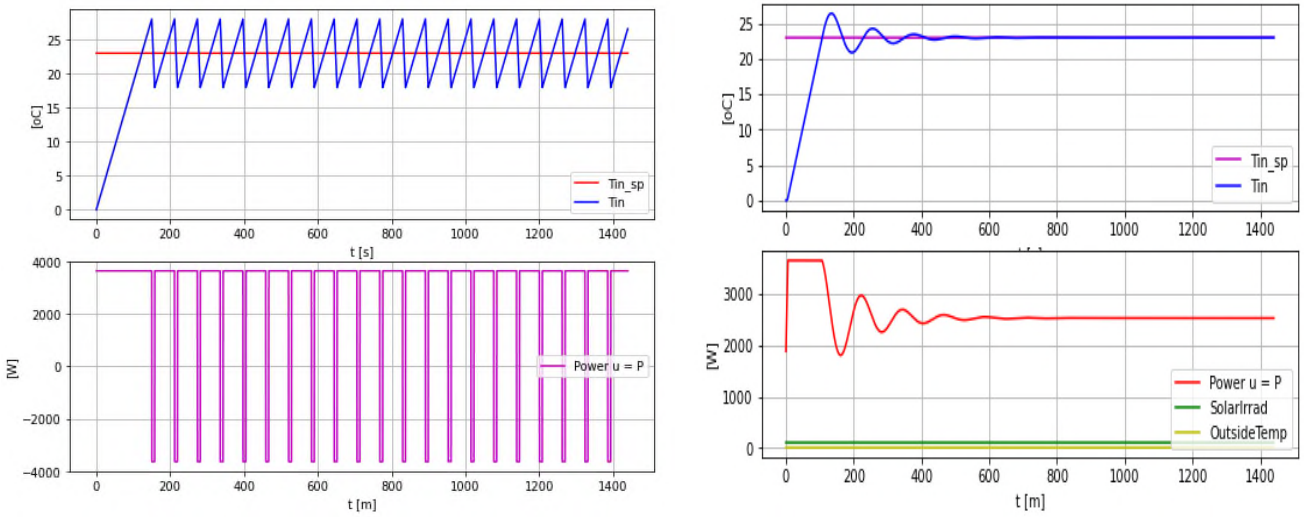
The controller evaluation is intended to assess the performance of the controllers in disturbance rejection and to determine the control algorithm that adequately maintains the grower's desire 23^0C. To achieve this, both controllers were given the same parameter values as shown in Table 4.3. The static value of solar radiation Q_{rad}, outdoor temperature T_{out}, the desired temperature set point T_{in_sp} and initial temperature $T_{in\_initial}$ was used in the simulation. The control energy input (heat) was limited to the derived static control input energy of 3640.2 (W) u_{max} and u_{min}= −3640.2 as the actuator limit (given in Equation 4.1) The Q_{rad} and T_{out} values were obtained from the outdoor disturbance data.

The remaining values used in the simulation for the On-off and PI controller were given below in Table 4.3.

Table 4.3 Trial and error PI and On-off control parameters

Parameters	Values
Q_{rad}	108.2
T_{out}	10.5
T_{in_sp}	23
$T_{in\_initial}$	0 °C

Figure 4.3 a, b depicts the result of the On-off and PI controller, with a static input disturbance value.



(a) (b)

Figure 4.3 (a) On-off control output; (b) PI control output

Figure 4.3 a, b above, shows the outcome of the On-off controller and the PI controller, respectively, during a 24-hour simulation period (Codes are in Appendix 4.3 and 4.4). The On-off control is based on a simple rule that turns the heating system on and off based on the difference between the desired temperature and the actual temperature. As depicted in Figure 4.3 a, the On-off controller was employed to regulate the indoor temperature. Although both controllers achieved the targeted indoor temperature, the On-off controller had a non-zero control error. On the other hand, the PI controller, which was tuned

through a trial-and-error approach, resulted in oscillatory and marginally delayed closed-loop responses during the transient phase, which eventually yielded stable results. To compare the energy-saving potential of the two controllers, Figure 4.3 a, b shows that the average control input (heat) required to regulate the temperature for the two controllers is nearly the same. However, the PI controller shows superior performance in terms of control error and oscillations.

In the next design, the simulation procedure for tuning the PI controller was discussed. Tuning the PI control is intended to achieve better control performance in comparison to the two controllers designed above.

4.2.2 Tuning the PI Controller

Internal model control (IMC) tuning for PI and PID controller:

Several strategies exist for tuning PID controllers, depending on the knowledge of the controlled system. Some tuning approaches rely on the process model, and controller settings can be derived from the model's parameters based on certain criteria, such as good rising time and settling time. One such technique is Internal Model Control (IMC), which incorporates the process model into the controller implementation (Horatiu & Andreescu, 2012). IMC is a popular model-based control method in the process control industry due to its simplicity, robustness, and excellent control performance (Chechare et al., 2017). IMC is also the most common tuning correlation for a PID controller. It extends the lambda tuning method by considering time delays.

In this subsection, the PI controller was tuned following the six-step process described in (Cooper, 2005), which was summarised into three steps.

- First, the controller output was simulated to approach the design level of operation and the data were recorded as process responses shown in Figure 4.2.
- Second, the process data was analysed to estimate the process gain (K_p), process time constant (τ_p), and process dead time (θ_p) using a first order plus dead time (FOPDT) model given by equation 4.2 below:

$$\tau_p \frac{d_{Tin(t)}}{dt} = -Tin(t) + K_p U(t - \theta_p) \quad 4.2$$

- Lastly, the resultant FOPDT model was used to obtain the initial tuning parameters for the PI controller given in Table 4.4.

The FOPDT model parameters estimated from the process data were used to generate initial estimations of the controller tuning parameters based on a correlation. The IMC tuning correlations with moderate tuning (Rice, 2010) were then applied to obtain the PI controller parameters. The moderate tuning involves setting τ_c to be the larger of 0.1τ_p or 0.8 θ_p i.e., $\tau_c = max(0.1\tau_p, 0.8\theta_p)$:

$$K_c = \frac{1}{K_p} \frac{\tau_p}{(\theta_p + \tau_c)} \quad 4.3$$

$$T_i = \tau_p \quad 4.4$$

where τ_c is the closed loop time constant, K_c is the proportional gain and T_i is the integral time constant. Table 4.4 below shows the resulting tuned controller parameters.

Table 4.4 Tuned PI controller parameters.

Parameters	Values
K_c	132.4
τ_i	151.9

The above controller parameters were used as the new PI algorithm controller constants and the resultant response is given in Figure 4.4 below. (Code is in Appendix 4.5)

The On-off controller is known for its simplicity and the fact that it does not require tuning, which makes it a popular choice for many applications. However, its inherent sustained oscillations can be a drawback. In this first set of controller comparison section, a comparison between the On-off controller and the tuned PI controller (as shown in Figure 4.4) revealed that the PI controller offers smoother control, making it the preferred option for the greenhouse. The tuned

PI controller not only provides excellent control with zero steady-state control error but also offers good system stability when the controller gains, and integral time are appropriately tuned. In cases where heuristic tuning methods are required, appropriate values should be selected to ensure a good result. It is worth noting that the average steady-state control input signal (heat) of the two controllers is almost identical even with the tuned controller.

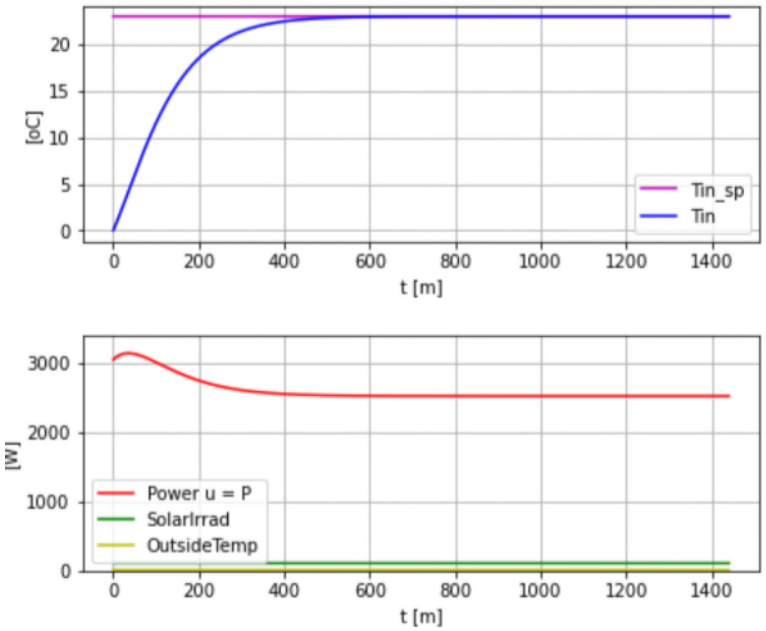


Figure 4.4 Tuned PI control output

In conclusion, two control algorithms, an on-off control, and a tuned PI control, were implemented to regulate the greenhouse's air temperature. While both controllers maintained the desired indoor temperature in simulation, the on-off controller had a non-zero mean control deviation from the setpoint. In contrast, the tuned PI controller offered smoother control and was selected as the better option. However, the on-off controller's low cost makes it a simple and practical control design for a small- or medium-sized greenhouse IoT control system solution. The simulation results confirmed that both controllers required the same steady-state control input signals (heat) to maintain the desired indoor temperature.

In the next section, a second set of controllers will be implemented and evaluated to assess their energy-saving potential at different setpoints. The goal is to

identify the best controller that minimises energy consumption while maintaining the desired indoor temperature.

4.3 On-off, PI, PID and MPC controller design

In this section, a suitable control strategy between on-off, PI, PID, and MPC control is determined. The simplified physics-based model of the system is used to evaluate the control systems. The control problem faced in this section is the optimisation of the four controllers to achieve energy minimization while maintaining a temperature variation range that is tolerable for the tomato (Solanum lycopersicum) plant in the prototype greenhouse. To achieve this objective, the setpoint was set to vary between 21 °C and 23 °C, rather than the precise temperature setpoint used in previous controllers. The performance of the optimised On-off controller is compared with that of the PI, PID, and MPC controllers using optimisation methods for disturbance rejection and energy minimization.

The On-off controller switches the heating element on and off based on a pre-set temperature threshold. In this simulation, the temperature hysteresis (a minimum and maximum temperature at which the heat will be turned on and off) was set to 0.5 and its performance was evaluated maintaining the temperature within a set range of 21°C to 23°C and minimizing energy.

The PI and PID controllers were tuned using the IMC method discussed in Section 4.2.2 The tuning process involved finding the appropriate tuning parameters that could minimise a performance criterion. Following the three steps discussed in Section 4.2.2, the analysed data was used to acquire the FOPDT model parameters (K_p), (τ_p), and (θ_p), and they were applied in Python to obtain the initial controller tuning parameters for the PI in Table 4.4, and PID in table 4.5 below.

Table 4.5 Tuned PID controller parameters.

Parameters	Values
K_c	176.179
τ_i	157.918
τ_d	5.7046

However, when these PID parameters were used, the optimizer could not find a solution. Therefore, the parameters were manually retuned, and the new values are presented in Table 4.6.

Table 4.6 Tuned PID controller parameters.

Parameters	Values
K_c	132.416
τ_i	15199171.36
τ_d	5.0

These retuned parameters were then used to obtain the PID controller output.

The MPC algorithm is designed to take system dynamics and constraints into account to optimise the control action. The MPC controller is a model-based control system that uses the simplified physics-based model of the system to predict the future behaviour of the system and determine the optimal control inputs.

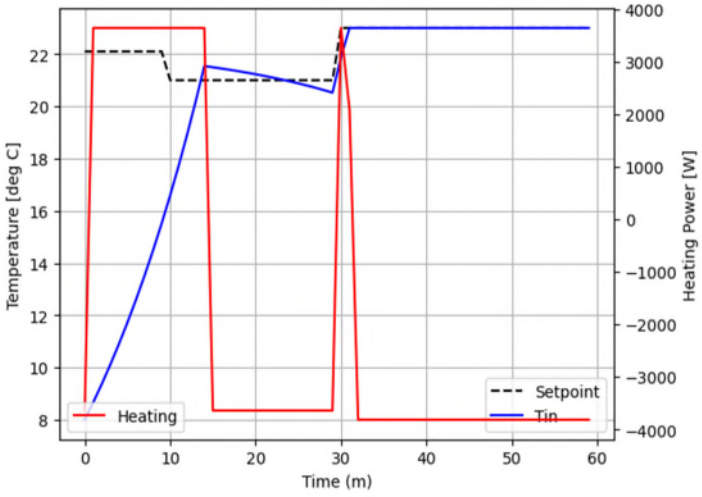
4.3.1 Controller design methodology

Methodology used for optimising the four controllers in Python using Gekko is describe below: Gekko is a Python-based optimisation software package that is particularly well-suited for solving dynamic optimisation problems. It includes a variety of powerful optimisation algorithms and modelling tools, to work on a wide range of optimisation problems (Beal & Hedengren, 2022) in areas such as process control, energy systems, and environmental engineering.

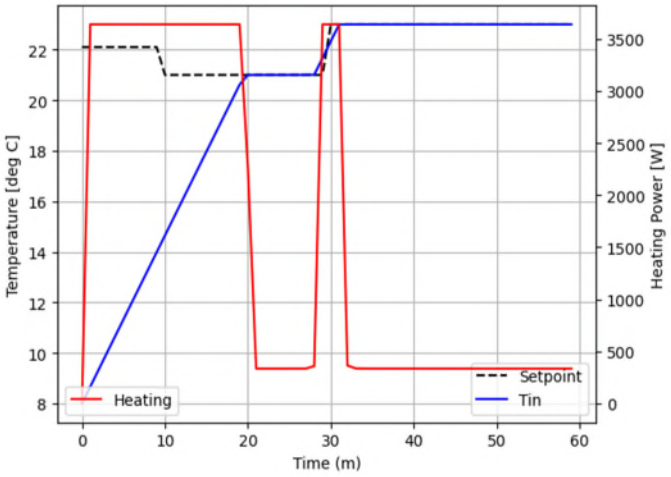
- First, the simplified physics-based model of the system developed in chapter 3 is used to evaluate the performance of the different control systems. The greenhouse parameters and control inputs are defined, and the process model is developed using Gekko. (Appendix 4.5 - 4.9),
- The control problem is defined as the optimisation of the four controllers (on-off, PI, PID, and MPC) to achieve a temperature variation range that is tolerable for the tomato (Solanum lycopersicum) plant in the prototype greenhouse. The setpoint is set to vary between 21°C and 23°C, rather than a precise temperature setpoint, to achieve two goals: first, to guarantee a required temperature that will not compromise plant growth, and second, to reduce (minimise) the control input energy of 3640.2 (W) u_{max} and u_{min} = −3640.2.
- Each controller is optimised separately using Gekko's optimisation capabilities for disturbance rejection and energy minimization. The objective function for all controllers is to minimise the difference between the setpoint and the controlled variable while minimising the energy used to maintain the temperature in an acceptable range.
- The performance of each optimised controller is compared to the others, and the results are analysed to determine which controller is most effective in achieving the desired temperature range while minimising energy use.
- Finally, the results are presented, and conclusions are drawn regarding the effectiveness of each controller in controlling the temperature of a greenhouse and their energy minimization capabilities.

4.3.1 Controller simulation results

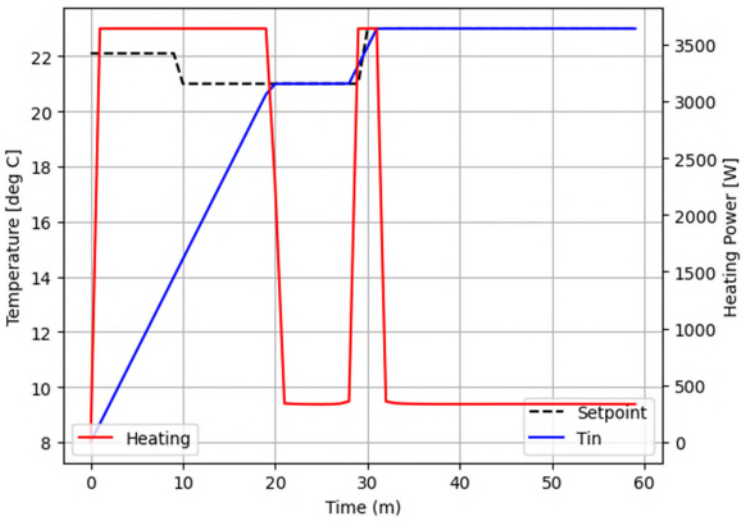
Figure 4.5 a, b, c, and d depict the results of the On-off and PI, PID and MPC controllers for setpoint tracking and energy minimization.



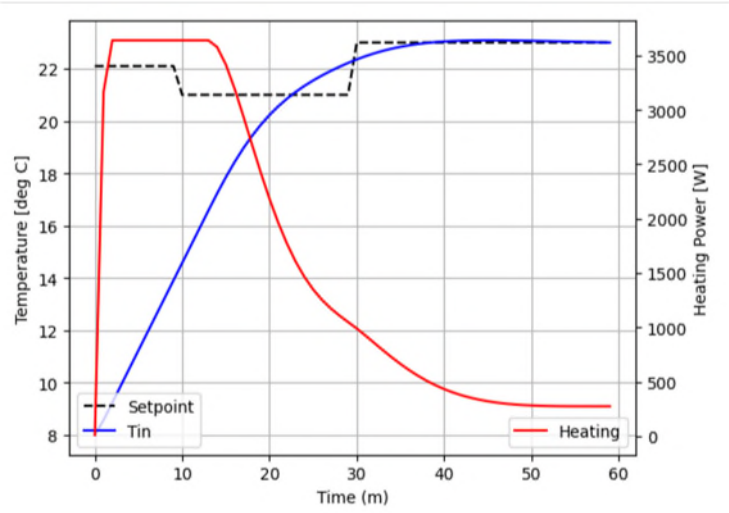
(a)
On-off controller



(b)
PI controller



(c)
PID controller



(d)
MPC controller

Figure 4.5 (a, b, c, d) Controller's performance

The controllers' performance (On-off, PI, PID, and MPC) was compared based on their ability to minimise energy consumption and track the set point. The implementation of all controllers used Gekko and the physics-based model. To determine which controller minimises heating and cooling energy the most, the

total amount of energy consumed by each controller was compared over the 60-minute simulation. The total energy consumed was calculated by multiplying the average power consumption for the duration of the simulation, which is 60 minutes.

The results of the simulation show that the On-off controller had the lowest average power consumption at 944.68 W, while the PI, PID, and MPC controllers had average power consumptions of 1574.12 W, 1574.09 W, and 1554.61 W, respectively. It should be noted that the data for all controllers, except for the vent signal, were recorded in the same way. The energy consumption was recorded based on the heating signal and not the vent signal. This is because opening the vent uses very little energy in a real sense, so it is not integrated into the energy consumption.

According to the study, the On-Off controller used the highest energy with an average power consumption of 944.68 watts. This is because the On-Off controller uses a fixed amount of power, which can result in overshooting or undershooting of the desired temperature, leading to energy wastage. On the other hand, the PI, PID, and MPC controllers adjust the power based on the difference between the current temperature and the desired setpoint, which can help reduce energy wastage due to proper tuning. The study found that the MPC controller had the lowest energy consumption overall, followed by the PID and PI controllers. The tracking error of the controllers was evaluated using the root mean square error (RMSE) between the set point and the measured temperature. Regarding the tracking error, the On-off controller had the lowest RMSE of 4.513^{o}C, while the PI and PID controllers had an RMSE of 4.867^{o}C, and the MPC controller had an RMSE of 4.920^{o}C.

4.3.2 Chapter summary

In this section, a suitable control strategy between On-off, PI, PID, and MPC control is determined. The simplified physics-based model of the system is implemented to evaluate the control system. The control problem faced in this section is optimising the four controllers to achieve the plant's maximum tolerable temperature and enable temperature range variations, rather than a precise

temperature setpoint, as discussed previously. This approach uses optimisation techniques to achieve energy minimization. The performance of the optimised On-off controller is compared with that of the PI, PID, and MPC controllers using optimisation methods for disturbance rejection and energy minimization.

The implication of the finding is that an optimised control strategy based on the PID and MPC controllers can effectively control the temperature of a greenhouse within a tolerable range for plant growth while minimising energy consumption. The On-off controller had a larger peak control effort because it switches on and off based on the set point, whereas the other controllers adjust the heating power continuously.

Overall, the results suggest that the MPC controller may be the most suitable controller for this heating system, as it consumes less energy and provides relatively smooth control. However, the On-off controller may be suitable for situations where continuous control is not necessary, and switching the heating input off and on does not cause wear to the actuator. Additionally, it may be suitable for small- to medium-sized growers.

CHAPTER 5 IoT Control System

5.1 IoT System Architecture

The goal of the established cloud-based control system is to address the energy efficiency problem faced in agricultural greenhouses by implementing a monitoring and control system. The cloud-based control system is intended to maintain the greenhouse microclimate by constantly monitoring and controlling the indoor temperature. The cloud-based control system enables remote access from anywhere at any time, providing farmers with real-time data and control of the greenhouse, allowing them to maintain the required temperature suitable for crops and gain insights to make informed decisions. The implementation of this system can lead to significant energy savings, increased crop yields, and improved overall greenhouse performance.

The simple and efficient IoT cloud-based control system is shown in the schematic diagram in Figure 5.1 below, it consists of a sensing part, a controlling part, and an actuating part.

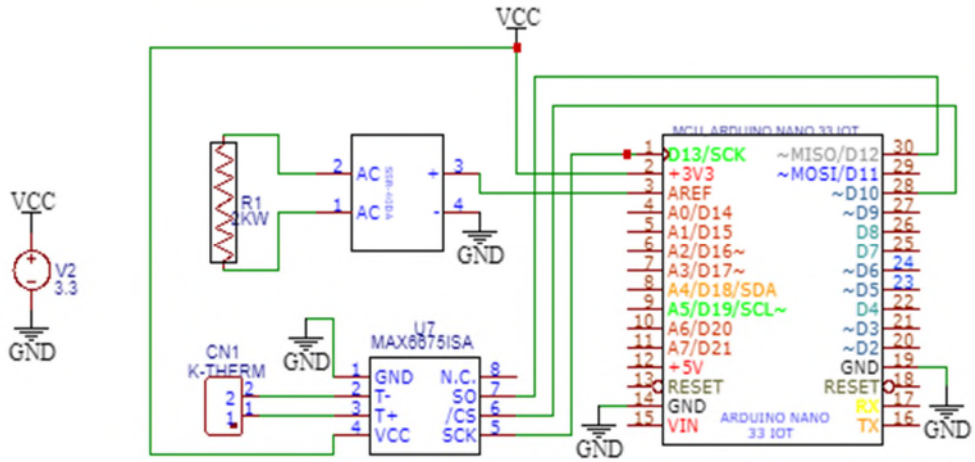


Figure 5.1 IoT-based control system schematic diagram

The content of the IoT cloud control system is shown in Figure 5.2 below.

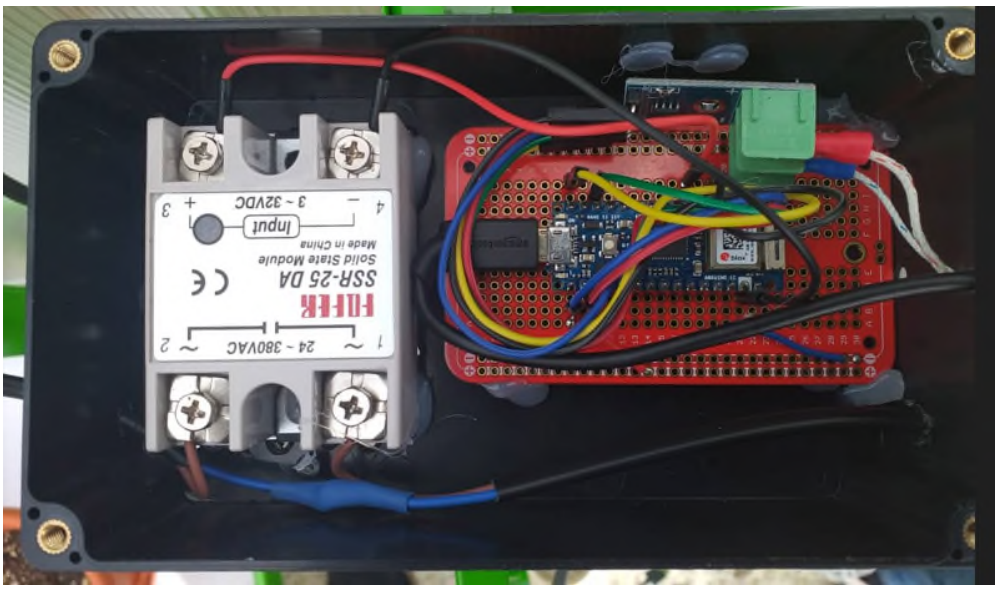


Figure 5.2 IoT-based control system content

Four of the control systems in total were deployed in the prototype greenhouse shown in Figure 5.3, this is the greenhouse that was used for this study. The greenhouse is located at Cranfield University, United Kingdom (52° N, 0° E), it has a floor area of 12.5 m^2 and consists of three green planting racks, the detailed description of the greenhouse is given chapter 6. one of the control systems was placed in the upper middle green rack and three others placed in the lower racks. The main units of the established IoT cloud-based control system are as follows:

- Arduino Nano 33 IoT microcontroller.
- K-type thermocouple sensor connected to the microcontroller.
- A Wi-Fi connection.
- Heater and electric bulbs.
- Arduino IoT cloud platform.
- User interface.

Four of the IoT-based control systems were deployed in the greenhouse as previously described. Each of them starts operation by activating the sensor

attached to the Arduino 33 IoT microcontroller. The K-type thermocouple sensor is positioned to measure the air (microclimate) temperature. The sensor reports its collected data to the microcontroller in real-time. Consequently, the microcontroller directly transmits the collected data to the Arduino IoT cloud platform through the phone that was turned into a Wi-Fi hotspot. The user gets access to the real-time data at any time of the day or in real-time, either through the Arduino IoT Cloud platform via the mobile app Figure 5.4 (a) or through the computer web-based user interface as shown in Figure 5.4 (b).

Figure 5.3 Experimental greenhouse.

Regarding the real-time decisions, a code was written (Appendix 5.1) to read data from the sensor and to control the heater in real-time. This code was written using Arduino Integrated Development Environment (IDE) and modified using the Arduino Create web editor, and the sketch was uploaded directly to the Arduino microcontroller from the web browser. The real-time decisions are then made by the microcontroller based on the software that is running on the Arduino microcontroller.

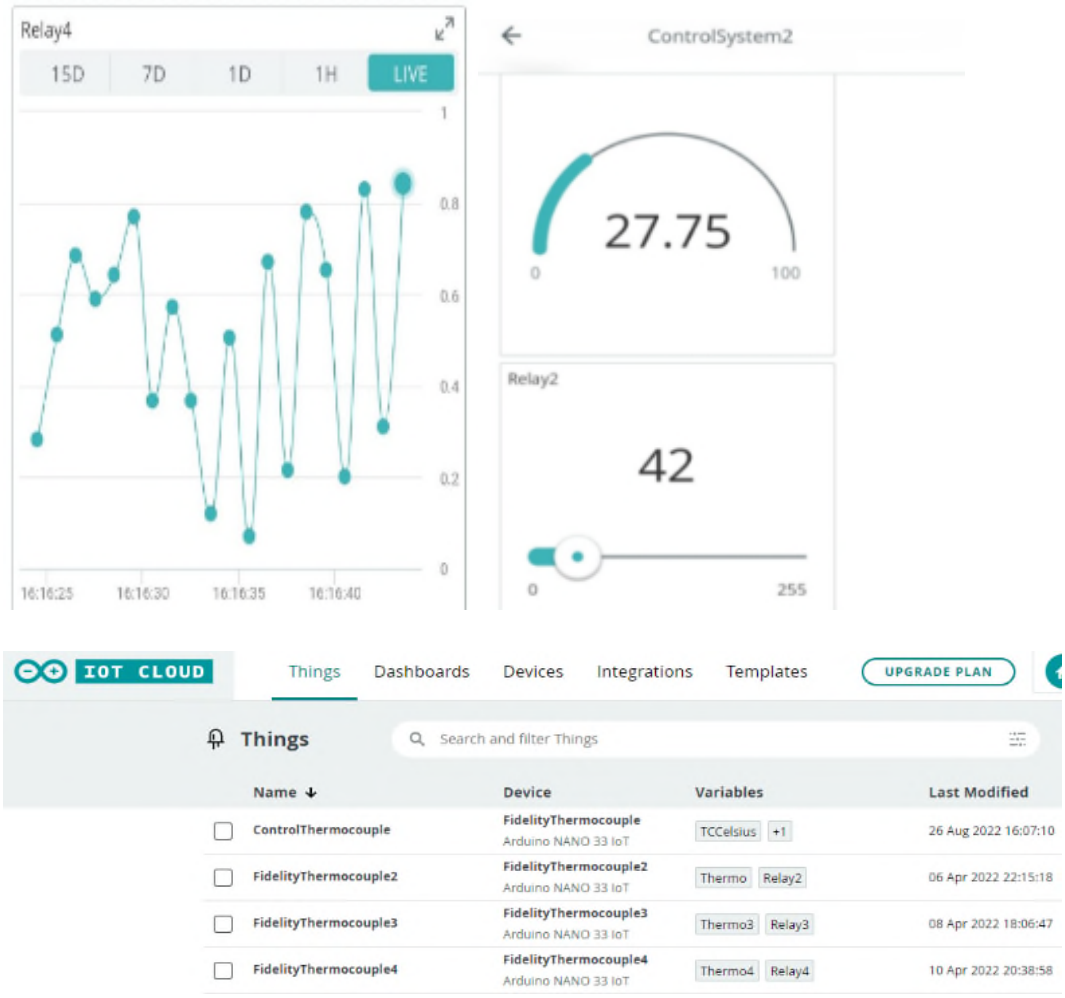


Figure 5.4 Arduino IoT cloud (a) phone App data display (b) Web based platform

5.1.1 Schematic Diagram Design

The control system schematic diagram in Figure 5.1 was designed using EasyEDA online design tool. EasyEDA is a web-based circuit design, circuit simulator, and PCB design tool that is free and simple to use. The IoT-based control system schematic diagram was designed using this tool.

5.1.2 Hardware Design

The aim of the developed IoT control system is to provide a simple and efficient solution for greenhouse monitoring and control. When designing the control system, one of the challenges faced was finding a source to power the IoT control

system, to avoid the know challenge of energy need requirement of IoT devices (Ramakrishnan et al., 2020), the choice was made to consider the available mains (220V) AC power supply near the greenhouse location. This constraint limited the choice of microcontrollers to those that could operate using a 220V AC power supply.

For this reason, the Arduino Nano 33 IoT microcontroller board was chosen to manage the sensor and actuator that were placed in the greenhouse. The Nano 33 IoT board is a low-power microcontroller with integrated Wi-Fi, Bluetooth, an inertial measurement unit (IMU), and a real-time clock (RTC). It has the same pins and form factor as the previous 8-bit Nano board and consists of 22 digital input and output pins, including 8 analogue input pins and 14 digital pins, 11 of which can be used for PWM output. The board has one 5V, one 3.3 V, and two ground pins. As shown in Figure 5.5, to connect the sensor and actuator (relay) to the Arduino microcontroller, a total of 4 pins was used for the connections. This provides an opportunity to extend the scale of the control system in the future.

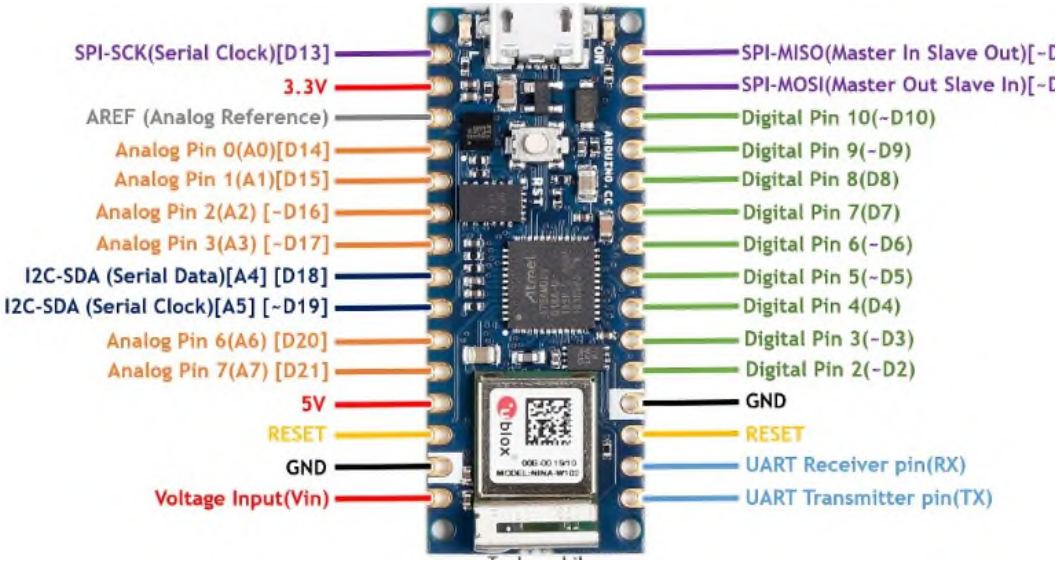


Figure 5.5 Arduino 33 IoT Nano pinout

MAX6675 in Figure 5.6 was used in the design of the control system. MAX6675 is an ADC that converts the K-type thermocouple's voltage output into a digital signal that the Arduino Nano 33 IoT board can read. The Arduino Nano 33 IoT board is compatible with the MAX6675, which operates at 3.3 V and has a

maximum output current of 50 mA, well within the maximum output current of the Arduino Nano 33 IoT board.

Figure 5.6 K-type thermocouple + MAX6675 module

A solid-state relay (SSR)**,** 40 An SSR-40DA relay shown in Figure 5.1 was chosen in the control system design. This solid-state relay can switch up to 40 A of 24–380 V AC loads. Although the maximum current that can be sourced by a single microcontroller pin is significantly less than 200 mA, the SSR is designed to operate directly from logic-level outputs. This simplifies the design, reduces the number of components, and eliminates the need for additional circuitry, making it ideal for switching power in industrial and household appliances. Nonetheless, if available, a simple transistor or optocoupler circuit will serve as a buffer between the microcontroller and the device.

The 2-kW heater shown in Figure 5.7 was used in the IoT-based control system that operates at 240 V AC and draws approximately 8.33 A of current. This falls within the rating of the SSR-40DA 40 A relay, making it suitable for the heater.

Figure 5.7 2KW Greenhouse heater

5.1.3 Software Design

A PID algorithm was developed using the Arduino Integrated Development Environment IDE, after the development, it was modified with Arduino Create web editor, the sketch was then uploaded to the Arduino microcontroller directly from the web. Figure 5.8 depicts a portion of the code that was developed in IDE.

The IoT cloud configuration procedure is described in detail below.

- The first step involves setting up a local laptop with an Arduino Create plugin. This plugin allows the Arduino 33 IoT board to communicate with the Arduino IoT Cloud via a micro-USB connection;
- The hardware setup in Figure 5.1 shows the connection of the Arduino 33 IoT board to the thermocouple sensor and the SSR;
- The board is then configured through the Arduino Create plugin mentioned in step 1. This process involves following simple instructions to register the device in the cloud and establish communication with the cloud via Wi-Fi;
- In the Arduino Cloud platform, the microcontrollers are referred to as “things.” Figure 5.4b shows the four connected microcontrollers called

"things." Each of these "things" has properties (a property is a cloud variable) that can be viewed and modified from the Arduino cloud platform;

```
// PID Parameters
double sensed_output, control_signal;
double setpoint;
double Kp = 2.04; //proportional gain
double Ki = 3.41; //integral gain
double Kd = 0.306; //derivative gain
int T = 50; //sample time in milliseconds (ms)
unsigned long last_time;
double total_error, last_error;
// Sensor parameters
int sensorPin = A5; // assign the analog input pin
int sensorVoltage; // the veriable storing the analog signal
double temperature; // sensed temperature
int OutputPin = 5;  // goes to SSR

void setup() {
  pinMode (OutputPin, OUTPUT);
  Serial.begin(9600);
```

Figure 5.8 Extract of the main code repeated on the control system

- For each of the microcontrollers, two properties are set up: TCCelcius or thermo (which represents temperature measured in degrees Celsius) and a relay (to vary the control input signal to the heater). The value of the temperature variables is set to range from 0–100 in the cloud, and the variable permission that is set is "Read Only" (this means the variable can only be read from the Arduino pin it is connected to but cannot be used to output voltage);
- The relay variable is set to range from 0–255 in the cloud, which is the maximum output that the Arduino PWM pin is allowed to output, and the variable permission that is set is "ReadWrite" (which means that it can both receive and send data to the Arduino pin that controls the relay. This allows the Arduino Cloud platform to modify the relay's control input signal and turn the relay on and off as needed).
- Finally, the data updating status is set for both the TCCelcius or thermo and relay variables to update data "on change," which means that data is updated as soon as it becomes available.

Although the temperature range was set on the cloud from 0–100, the PID algorithm running on the device is already programmed to a setpoint of 23 °C that

is required for the plant in this work. The control system input and output data are given in Figure 5.9.

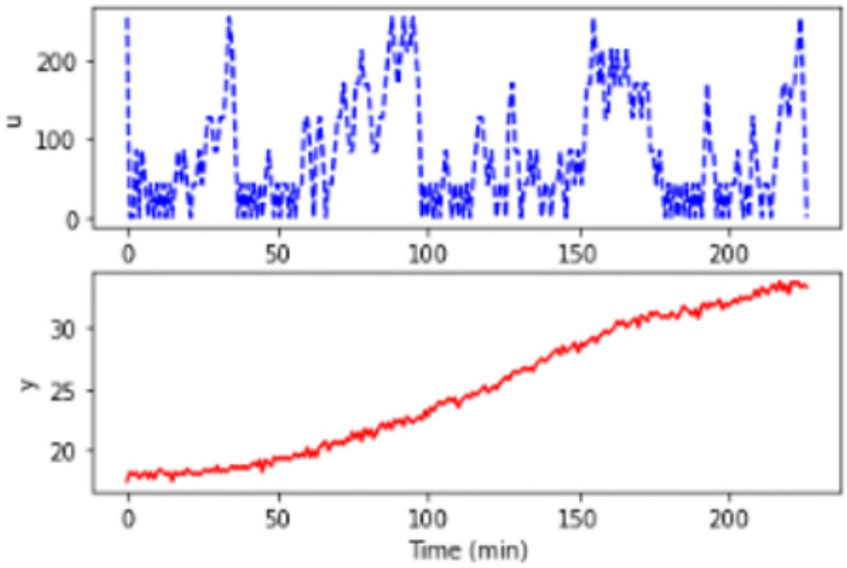


Figure 5.9 IoT-based control system measured input/output data

5.1.2 Control system performance result and discussion

In this section, the performance of the IoT control system deployed to tackle the energy efficiency problem in agricultural greenhouses was verified using data collected from the first and last experiment conducted on May 11, 2022, and May 25, 2022, respectively.

The IoT-based control system showed promising performance in maintaining the indoor temperature within the desired range. For the 11th data point in Figures 5.10, the system was able to maintain the indoor temperature around 28 degrees Celsius, with some fluctuations due to variations in heating input. The heating input was able to adjust and control the temperature within the desired range, indicating the effectiveness of the control system. However, there were some limitations in the data collection process, as the heating input data was only available for 299 data points, which is much less than the temperature data. This may have limited the accuracy of the control system and its ability to maintain the temperature within a narrow range.

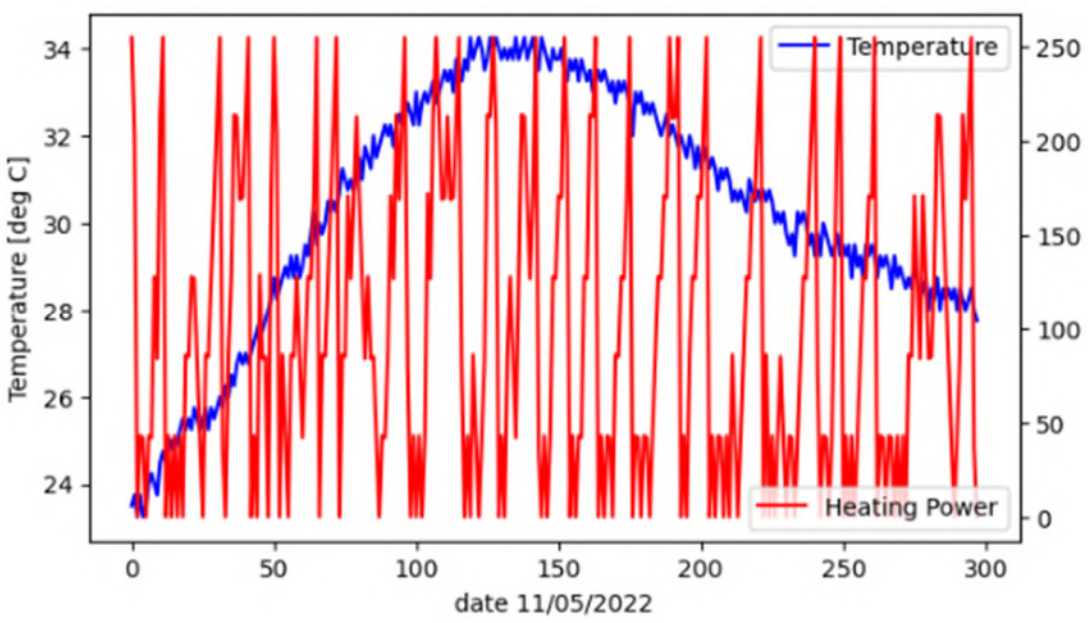


Figure 5.10 Performance tracking chart 1

For the 25th data point in Figures 5.11, the temperature was initially set to 28 degrees Celsius but was later changed to 23 degrees Celsius. The control system was able to adjust and maintain the temperature within the new desired range, indicating the adaptability and flexibility of the system. Furthermore, it is important to note that environmental factors like solar radiation and outdoor temperature had an impact on the temperature sensor and control system's performance. The vent installed in the greenhouse was wide enough to handle these disturbances, but they still impacted the measured temperature. Therefore, in future iterations of the IoT-based control system, additional measures can be taken to mitigate the impact of environmental disturbances on the performance of the system.

Overall, the performance of the temperature sensor, heater, and control system together was promising, as the system was able to maintain the indoor temperature within the desired range. However, to improve the system's accuracy and effectiveness, it may be necessary to ensure that the data collection process provides sufficient data for all variables, including heating input. Additionally, continuous monitoring and analysis of the system's performance can help identify areas for improvement and optimise its operation.

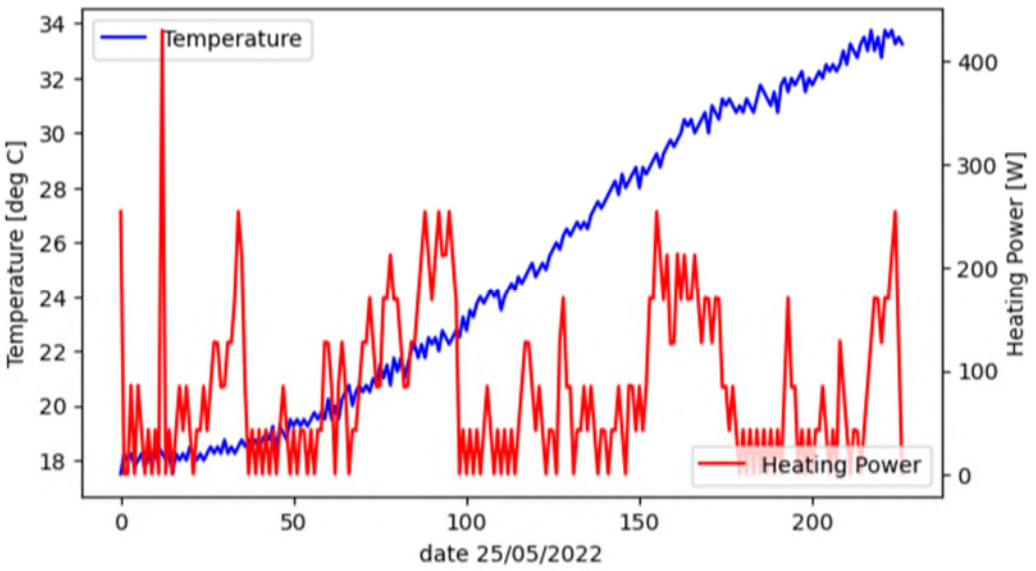


Figure 5.11 Performance tracking chart 2

The deployed IoT control system was designed to improve energy efficiency and reduce waste in the greenhouse heating system. By using a pulse width modulation (PWM) signal output, the control system regulates the power output of the greenhouse heater, i.e., varying to provide only the required amount of energy needed to maintain the desired temperature setpoint as plotted figures, subsequently, saving the energy use. This approach is more efficient than a traditional on-off controller, which would continuously switch the heating system on and off. Additionally, the control system is compact. While the on-off, MPC, and PI algorithms have not been tested on the system, the preliminary results show promising potential for further optimisation and energy savings.

5.1.2 Chapter summary

In conclusion, the implementation of the IoT-based control system resulted in numerous advantages, including automatic control of greenhouse heating and PID controllers designed to run on the microcontroller without external processing. These features make the system practical and suitable for computerising greenhouse management by monitoring and controlling plants' required temperatures. The system enables growers to adjust the setpoint to accommodate different temperature ranges suitable for different crops,

maximising planting space without affecting plant growth. However, the present work's drawback is that the greenhouse turning algorithm implementation was based on trial and error or heuristic turning techniques, and the set-point tracking was not very effective, as shown in Figure 5.11. The PID system's proper tuning was eventually developed in simulation, which was discussed in Chapter 4.

CHAPTER 6 Experimental study

6.1 Methodology - Greenhouse set-up

The image of the greenhouse that was built for this research is shown in Figure 6.1 below. The methodology for the experimental study on greenhouse temperature dynamics is described below in 6 steps.

Figure 6.1 Experimental greenhouse

1. Description of the greenhouse structure and equipment used:

The greenhouse used in this study was a 4ft by 6ft prototype greenhouse that is suitable for growing tomatoes from seed. It was built on an open site at Cranfield University. The greenhouse had a floor area of 12.5 m^2 and a roof consisting of a single aluminium frame vent mounted at a slope of 60°. The greenhouse was equipped with a 2KW heater (placed in the upper middle green rack) shown in Figure 6.2, artificial lighting shown in Figure 6.3 **Error! Reference source not found.**, natural ventilation shown in Figure 6.4 **Error! Reference source not found.**, and three racks consisting of two 50W bulbs and one 25W bulb each, one of them is shown in Figure 6.5 below. These heating bulbs were used as an additional heating system to maintain the desired microclimate temperature on the lower racks on the greenhouse floor.

Figure 6.2 2KW Greenhouse heater

Figure 6.3 Greenhouse artificial lighting

Figure 6.4 Natural ventilation

Four of the IoT-based control systems discussed in Chapter 5 were deployed in the greenhouse to monitor the microclimate around each of the three racks. Each

of the built IoT control systems consisted of one MAX6675 module + K-type Type Thermocouple sensor, either a 50W bulb, a 25W bulb, or the 2KW heater, and one Arduino Nano 33 microcontroller. The sensors were positioned in such a way that they could measure the temperature variation in the immediate environment of the plant placed in the greenhouse rack.

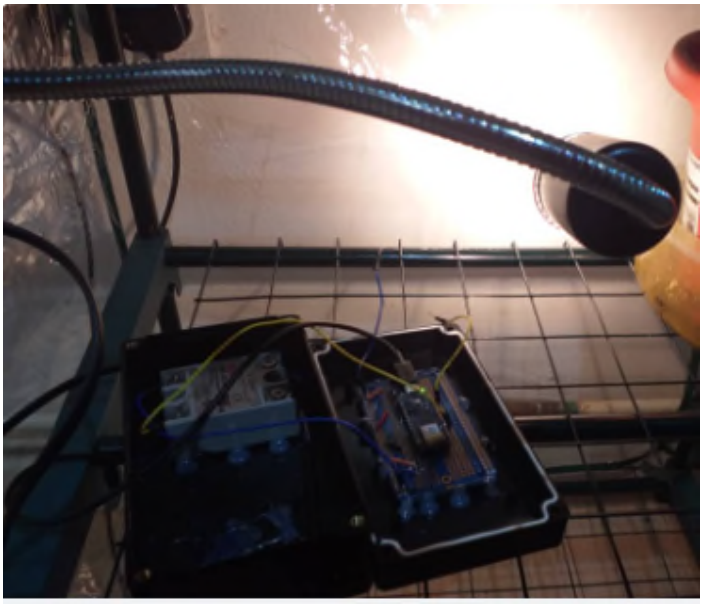


Figure 6.5 Additional heating bulbs

1. Environmental conditions:

During the study, the environmental conditions inside the greenhouse were maintained at specific levels. The temperature was controlled using the heating system, which consisted of a 2 KW heater and three sets of heating lamps. The heating system consisted of 4 heaters in total with 4 sensors; 3 of the heaters (heating lamps) were placed in each of the racks at a height of approximately 0.32m and the sensors at 0.26m shown in the ground, with a distance of 0.04 between both of them as shown in Figure 6.6 below. The fourth heater shown in Figure 6.7 was placed on the upper layer of the centre rack to measure the greenhouse air temperature. The greenhouse was also equipped with natural ventilation provided by the roof vent, which was used to air the greenhouse as a switch actuator every 1 hour and 6 minutes for 5 minutes until the afternoon.

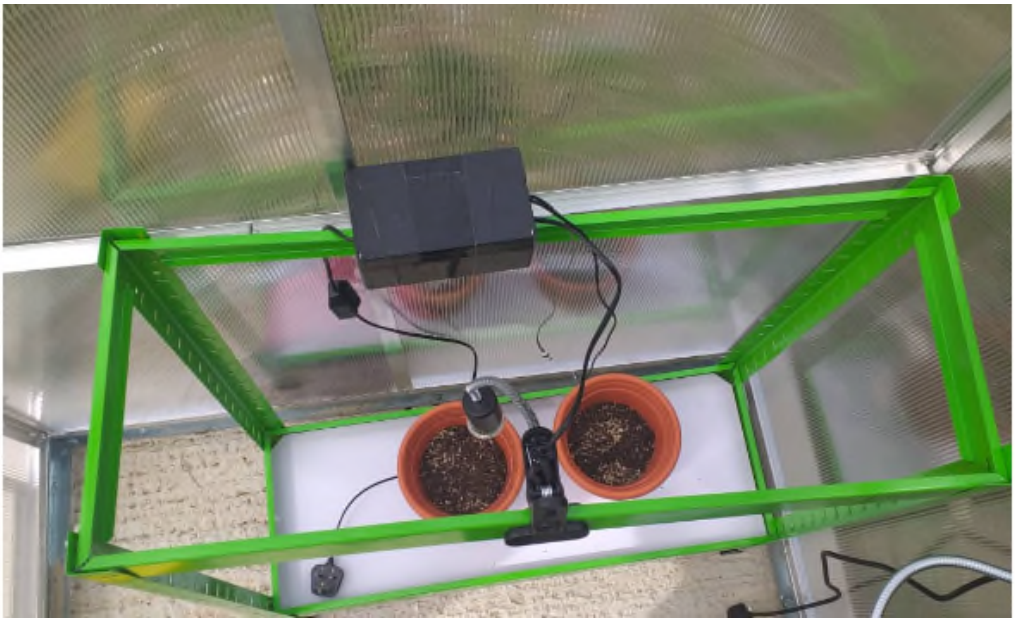


Figure 6.6 Heater and sensor placement spacing distance

2. Experimental design:

The overall design of the study involved conducting three sets of experiments from 11th May 2022 until 13th May 2022, from 18th May 2022 until 19th May 2022, and on 25th May 2022. The study aimed to measure the temperature dynamics in greenhouse air for calibration and validation of the built models discussed in Chapter 3. Tomato plants (Solanum lycopersicum) were sown inside pots in a mixture of vermiculite compost, lightweight container, and basket compost, and Miracle-Gro all-purpose water-soluble plant food. However, no further experiment or analysis was conducted on the plant, as it was not the present focus of this work.

Figure 6.7 Greenhouse centre sensor

3. Data collection methods:

Data was collected using the four IoT-based control systems that were deployed in the greenhouse to monitor the microclimate around each of the three racks and sent to the Arduino IoT cloud platform discussed in Chapter 5 for logging, analysis, visualisation, and control purposes. Each of the built IoT control systems is shown in Figure 6.8 below. consisting of one MAX6675 Module + K-type Type Thermocouple sensor, either a 50W bulb, a 25W bulb, or the 2KW heater, and one Arduino Nano 33 microcontroller. The sensors were positioned in such a way that they could measure the temperature variation in the immediate environment of the plant placed in the greenhouse rack. The data was collected continuously during the experiments, and the frequency of data collection was approximately 40 seconds, as set in the instruction in the Arduino IoT cloud to update data “on change,” as described in Chapter 5.

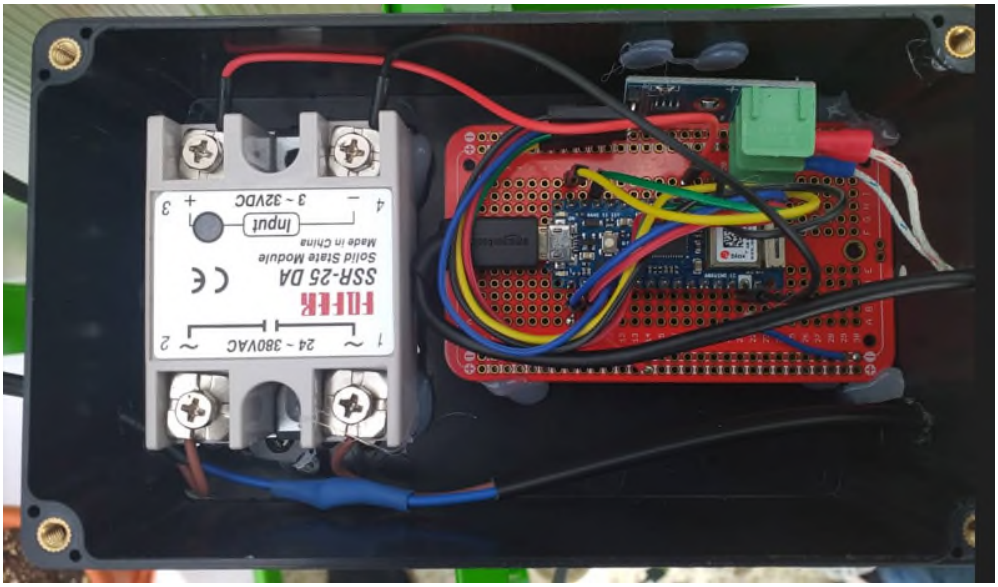


Figure 6.8 The control system content

4. Statistical analysis:

The data collected during the experiments were analysed using statistical methods, for example, descriptive statistics such as mean absolute error (MAE), mean squared error (RMSE)**,** and R-squared (R2), regression analysis and Analysis of Variance (ANOVA)**.** The statistical analysis aimed to calibrate and validate the greenhouse temperature dynamic model.

5. Limitations and assumptions:

One of the primary limitations of the study was the size of the greenhouse used in the experiment. While the 4ft by 6ft prototype greenhouse was sufficient for the purposes of this study, it may not accurately represent the temperature dynamics of larger or more complex greenhouse structures. Additionally, the study only focused on measuring temperature dynamics in greenhouse air and did not consider other factors that may affect plant growth and development, such as soil moisture, nutrient levels, or pest and disease pressure.

Another potential limitation of the study was the use of IoT-based control systems for monitoring temperature dynamics in the greenhouse. While these control systems were effective for the purposes of this study, they may not be suitable for other greenhouse environments that have different heating and ventilation

systems or other variables that may affect temperature dynamics. It is important to consider these limitations and assumptions when interpreting the results of the study and applying them to other greenhouse systems or environments.

6.1.1 Chapter summary

Chapter 6 focuses on the experimental study that was conducted to measure temperature dynamics in greenhouse air for calibration and validation of built models. The chapter begins by describing the greenhouse structure and equipment used for the study, including details about the size of the greenhouse, and the equipment used to maintain the greenhouse environment. The chapter then goes on to describe the environmental conditions that were maintained inside the greenhouse during the study, namely the temperature.

Next, the chapter discusses the experimental design and conditions. The chapter also covers the data collection methods used during the study, including the type of sensors used, the frequency of data collection, and the methods used to store and analyse the data.

The chapter list some of the statistical analysis used to analyse the data collected during the study, including descriptive statistics, regression analysis, and ANOVA. The chapter concludes with a discussion of the limitations and assumptions of the study.

CHAPTER 7 Conclusion

This concluding chapter reflects on the research process, limitations, and potential for future development.

A. Summary of the Research Question and Methods

The study aimed to investigate how an automated control system based on Internet of Things (IoT) technology could optimise energy use in greenhouse systems. To achieve this, four controllers were tested: MPC, PID, PI, and On-Off controller. The controllers were evaluated based on their ability to maintain a target temperature of 23°C in a simulated greenhouse environment while minimising energy consumption. Disturbance rejection, setpoint tracking, and energy savings achieved relative to the On-Off controller were used to compare the controllers. The research methods included experimentation, mathematical modelling, simulation, and data analysis.

B. Summary of the Main Findings

This thesis aims to optimise energy use in greenhouse systems using an IoT-based control system. A greenhouse prototype system was built, and four Arduino-based control systems were deployed. Two control algorithms were used to track the reference temperature and maintain a comfortable environment for crops, while four were used to verify energy use minimization. Physics-based and FOPDT models were developed to optimise energy use in the greenhouse. The physics-based model was found to be a better predictor of indoor temperatures than the FOPDT model.

Energy-saving performance was evaluated for all four controllers, with the four controllers evaluated in terms of their ability to maintain a target temperature of 23°C in a simulated greenhouse environment while minimising energy consumption. A simulation study was conducted to compare the performance of four different controllers - On-off, PI, PID, and MPC - for a heating system. The controllers were evaluated based on their power consumption, control effort, and tracking error. The simulation was run for 60 minutes with varying set points, and

the control input signals were limited to a minimum heating power of -3640.2 W and a maximum heating power of 3640.2 W.

The results of the simulation show that the On-off controller had the lowest average power consumption at 944.68 W, while the PI, PID, and MPC controllers had average power consumptions of 1574.12 W, 1574.09 W, and 1554.61 W, respectively. It should be noted that the data for all controllers, except for the vent signal, were recorded in the same way. The energy consumption was recorded based on the heating signal and not the vent signal. This is because opening the vent uses very little energy in a real sense, so it is not integrated into the energy consumption.

According to the study, the On-Off controller used the highest energy with an average power consumption of 944.68 watts. This is because the On-Off controller uses a fixed amount of power, which can result in overshooting or undershooting of the desired temperature, leading to energy wastage. On the other hand, the PI, PID, and MPC controllers adjust the power based on the difference between the current temperature and the desired setpoint, which can help reduce energy wastage if tuned properly. The study found that the MPC controller had the lowest energy consumption overall, followed by the PID and PI controllers.

The On-off controller had a larger peak control effort because it switches on and off based on the set point, whereas the other controllers adjust the heating power continuously. The tracking error of the controllers was evaluated using the root mean square error (RMSE) between the set point and the measured temperature. Regarding the tracking error, the On-off controller had the lowest RMSE of 4.513^{o}C, while the PI and PID controllers had an RMSE of 4.867^{o}C, and the MPC controller had an RMSE of 4.920^{o}C.

Overall, the results suggest that the MPC controller may be the most suitable controller for this heating system, as it consumes less energy and provides relatively smooth control. However, the On-off controller may be suitable for situations where continuous control is not necessary, and switching the heating

input off and on does not cause wear to the actuator. Additionally, it may be suitable for small- to medium-sized growers.

C. Implications of the Findings

• Improved energy efficiency in greenhouse systems: The MPC and PID controllers resulted in significant energy savings, suggesting that commercial greenhouse operators can benefit from advanced control systems to reduce energy costs and promote sustainability. The On-off controller remains a viable option for small to medium-sized growers.

• Importance of accurate modelling: The physics-based model was a better predictor of indoor temperature and can be adapted for use with other IoT-based system modelling.

• Generalizability of the IoT-based control system: The system can be applied to other greenhouse systems with similar control systems and actuators, potentially leading to widespread adoption of energy-efficient and sustainable greenhouse systems.

D. Limitations

The limitations of the study include the use of a small-scale greenhouse prototype, which may not fully capture the complexity and variability of larger-scale commercial greenhouse systems. Additionally, the study only focused on optimising the microclimate temperature of the greenhouse without considering other important factors such as humidity and CO2 concentration. Furthermore, the study did not account for the effects of crop type on energy use and temperature control. This study is an interdisciplinary approach to solving complex problems in agricultural practises and developing more efficient and cost-effective greenhouse heating systems. However, further findings are needed to verify its applicability to larger-scale commercial greenhouse systems and specific crops.

E. Discussion on challenges that affect the work and result.

it is clear that the project was faced with several challenges that affected the timeline and the scope of the study.

The first major setback was the destruction of the initial greenhouse prototype by a severe storm. Additionally, the outbreak of the Covid-19 pandemic caused delays and restrictions to the resources needed for the experimental study. Furthermore, there were delays in the supply of materials, which hindered the workflow. These difficulties made it challenging to gather enough data for a more in-depth analysis and resulted in some limitations in the findings.

F. Suggestions on ways to address the challenges in future.

To address these challenges in future research, it may be helpful to plan for contingencies, such as having backup facilities or materials in case of unforeseen events. Additionally, exploring alternative methods of data collection, such as remote monitoring and control, could reduce the need for physical presence at the research site and mitigate the effects of any external disruptions. Collaboration with other researchers and institutions could also provide access to additional resources and expertise. Finally, a flexible timeline that accounts for potential delays should be considered during the planning phase to ensure the research can be completed within a reasonable timeframe.

G. Suggestions for future work

1. Research could focus on the development of more comprehensive greenhouse energy management systems that consider multiple environmental factors such as humidity and CO2 concentration. Additionally, the study can be expanded to include larger-scale commercial greenhouse systems to improve the generalizability of the findings. Future research can also include the effect of different crops on energy use and temperature control to develop crop-specific energy management strategies. Finally, the use of machine learning techniques can be explored to further optimise the control algorithms and improve energy savings.

2. The current study provides valuable insights into the use of different control algorithms for greenhouse temperature management and their impact on energy

efficiency. Future research can build on these findings by exploring the use of advanced control algorithms in combination with other environmental factors to develop more comprehensive greenhouse energy management systems. Additionally, the findings of this study can be used as a benchmark for evaluating the energy efficiency of different greenhouse systems and control strategies.

3. Future research can use a combination of experimental studies and simulation-based approaches to validate the four controllers' performances. Experimental studies can be used to validate the performance of different control algorithms in real-world conditions, while simulation-based approaches can be used to explore the impact of different environmental factors and control strategies on greenhouse energy efficiency. Additionally, the use of data-driven approaches such as machine learning can be explored to develop more accurate models of greenhouse energy consumption and temperature control.

REFERENCES

Abdel-Ghany, A. M., & Kozai, T. (2006). Dynamic modeling of the environment in a naturally ventilated, fog-cooled greenhouse. *Renewable Energy*, *31*(10), 1521–1539. https://doi.org/10.1016/j.renene.2005.07.013

Abdullah, A., & Barnawi, A. (2012). Identification of the Type of Agriculture Suited for Application of Wireless Sensor Networks. *Russian Journal of Agricultural and Socio-Economic Sciences*, *12*(12), 19–36. https://doi.org/10.18551/rjoas.2012-12.02

Abioye, E. A., Abidin, M. S. Z., Mahmud, M. S. A., Buyamin, S., AbdRahman, M. K. I., Otuoze, A. O., Ramli, M. S. A., & Ijike, O. D. (2020). IoT-based monitoring and data-driven modelling of drip irrigation system for mustard leaf cultivation experiment. *Information Processing in Agriculture*, *xxxx*. https://doi.org/10.1016/j.inpa.2020.05.004

Afzali, S., Mosharafian, S., van Iersel, M. W., & Velni, J. M. (2021). Development and implementation of an iot-enabled optimal and predictive lighting control strategy in greenhouses. *Plants*, *10*(12), 1–23. https://doi.org/10.3390/plants10122652

Ahamed, M. S., Guo, H., & Tanino, K. (2019). Energy saving techniques for reducing the heating cost of conventional greenhouses. *Biosystems Engineering*, *178*, 9–33. https://doi.org/10.1016/j.biosystemseng.2018.10.017

Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). A survey on sensor networks. *IEEE Communications Magazine*, *40*(8), 102–105. https://doi.org/10.1109/MCOM.2002.1024422

Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., & Ayyash, M. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys and Tutorials*, *17*(4), 2347–2376. https://doi.org/10.1109/COMST.2015.2444095

Alpár, G., Batina, L., Batten, L., Moonsamy, V., Krasnova, A., Guellier, A., & Natgunanathan, I. (2016). New directions in IoT privacy using attribute-based authentication. *2016 ACM International Conference on Computing Frontiers - Proceedings*, 461–466. https://doi.org/10.1145/2903150.2911710

Al-turjman, F. (2019). The road towards plant phenotyping via WSNs : An overview. *Computers and Electronics in Agriculture*, *161*(September 2018), 4–13. https://doi.org/10.1016/j.compag.2018.09.018

Aqeel-Ur-Rehman, Abbasi, A. Z., Islam, N., & Shaikh, Z. A. (2014). A review of wireless sensors and networks' applications in agriculture. *Computer Standards and Interfaces*, *36*(2), 263–270. https://doi.org/10.1016/j.csi.2011.03.004

Åström, K. J., & Hägglund, T. (2000). The Future of PID Control. *IFAC Proceedings Volumes*, *33*(4), 19–30. https://doi.org/10.1016/s1474-6670(17)38216-2

Aung, M. M., & Chang, Y. S. (2014). Temperature management for the quality assurance of a perishable food supply chain. *Food Control*, *40*, 198–207. https://doi.org/10.1016/j.foodcont.2013.11.016

Baccelli, E., Hahm, O., Gunes, M., Wahlisch, M., & Schmidt, T. (2014). *RIOT OS: Towards an OS for the Internet of Things*. 79–80. https://doi.org/10.1109/infcomw.2013.6970748

Bai, X., Wang, Z., Sheng, L., & Wang, Z. (2019). Reliable Data Fusion of Hierarchical Wireless Sensor Networks With Asynchronous Measurement for Greenhouse Monitoring. *IEEE Transactions on Control Systems Technology*, *27*(3), 1036–1046. https://doi.org/10.1109/TCST.2018.2797920

Bakker, J.C., Bot, G.P.A., Challa, H. V. de B. N. J. (1995). Greenhouse Climate Control an integrated approach. In J. C. Bakker, G. P. A. Bot, H. Challa, & N. J. van de Braak (Eds.), *Encyclopedia of Earth Sciences Series: Vol. Part*

4. Wageningen Academic Publishers. https://doi.org/10.3920/978-90-8686-501-7

Balamurali, R., & Kathiravan, K. (2015). An analysis of various routing protocols for Precision Agriculture using Wireless Sensor Network. *Proceedings - 2015 IEEE International Conference on Technological Innovations in ICT for Agriculture and Rural Development, TIAR 2015*, *Tiar*, 156–159. https://doi.org/10.1109/TIAR.2015.7358549

Bapat, V., Kale, P., Shinde, V., Deshpande, N., & Shaligram, A. (2017). Original papers WSN application for crop protection to divert animal intrusions in the agricultural land. *Computers and Electronics in Agriculture*, *133*, 88–96. https://doi.org/10.1016/j.compag.2016.12.007

Barrenetxea, G., Schaefer, G., & Vetterli, M. (2008). *Wireless Sensor Networks for Environmental Monitoring : The SensorScope Experience*. 98–101.

Beal, L., & Hedengren, J. (2022). *GEKKO Documentation Release 1.0.5*.

Bersani, C., Fossa, M., Priarone, A., Sacile, R., & Zero, E. (2021). *Model Predictive Control versus Traditional Relay Control in a High Energy Efficiency Greenhouse*. 1–21. https://doi.org/10.3390

Bersani, C., Ouammi, A., & Sacile, R. (2020). *Model Predictive Control of Smart Greenhouses as the Path towards Near Zero Energy Consumption*. 1–17.

Beschi, M., Pawlowski, A., Guzmán, J. L., Berenguel, M., & Visioli, A. (2014). Symmetric send-on-delta PI control of a greenhouse system. *IFAC Proceedings Volumes (IFAC-PapersOnline)*, *19*(D), 4411–4416. https://doi.org/10.3182/20140824-6-za-1003.01028

Bkheet, S. A., & Agbinya, J. I. (2021). A Review of Identity Methods of Internet of Things (IOT). *Advances in Internet of Things*, *11*(04), 153–174. https://doi.org/10.4236/ait.2021.114011

Blasco, X., Martínez, M., Herrero, J. M., Ramos, C., & Sanchis, J. (2007). Model-based predictive control of greenhouse climate for reducing energy and

water consumption. *Computers and Electronics in Agriculture*, *55*(1), 49–70. https://doi.org/10.1016/j.compag.2006.12.001

Boaventura Cunha, J., Couto, C., & Ruano, A. E. (1997). Real-time parameter estimation of dynamic temperature models for greenhouse environmental control. *Control Engineering Practice*, *5*(10), 1473–1481. https://doi.org/10.1016/S0967-0661(97)00145-7

Bontsema, J., Van Henten, E. J., Gieling, T. H., & Swinkels, G. L. A. M. (2011). The effect of sensor errors on production and energy consumption in greenhouse horticulture. *Computers and Electronics in Agriculture*, *79*(1), 63–66. https://doi.org/10.1016/j.compag.2011.08.008

Boulard, T., & Baille, A. (n.d.). *Modelling of AIr Exchange Rate in a Greenhouse Equipped with continuous Roof Vents* (p. 1995).

Bounaama, F., & Draoui, B. (2011). Greenhouse environmental control using optimized MIMO PID technique. *Sensors and Transducers*, *133*(10), 44–52.

Buratti, C., Conti, A., Dardari, D., & Verdone, R. (2009). An overview on wireless sensor networks technology and evolution. *Sensors*, *9*(9), 6869–6896. https://doi.org/10.3390/s90906869

Burhan, M., Rehman, R. A., Khan, B., & Kim, B. S. (2018). IoT elements, layered architectures and security issues: A comprehensive survey. *Sensors (Switzerland)*, *18*(9), 1–37. https://doi.org/10.3390/s18092796

Burn, K., & Cox, C. (2020). A hands-on approach to teaching system identification using first-order plus dead time modelling of step response data. *International Journal of Electrical Engineering and Education*, *57*(1), 24–40. https://doi.org/10.1177/0020720918813825

Cao, Q., Abdelzaher, T., Stankovic, J., & He, T. (2008). The LiteOS operating system: Towards Unix-like abstractions for wireless sensor networks. *Proceedings - 2008 International Conference on Information Processing in Sensor Networks, IPSN 2008*, 233–244. https://doi.org/10.1109/IPSN.2008.54

Capable, N., & Processor, A. (2010). *INTERNATIONAL STANDARD ISO / IEC / IEEE transducer interface for sensors and* (Vol. 2010).

Chalabi, Z. S., Bailey, B. J., & Wilkinson, D. J. (1996). Computers and electronics in agriculture A real-time optimal control algorithm for greenhouse heating. *Computers and Electronics in Agriculture*, *1*, 1–13.

Chechare, D., Kadu, C., & Parvat, B. (2017). *DESIGN OF INTERNAL MODEL CONTROLLER-BASED PID CONTROLLER* (Vol. 3). www.ijariie.com

Chen, J., Yang, J., Zhao, J., Xu, F., Shen, Z., & Zhang, L. (2016). Energy demand forecasting of the greenhouses using nonlinear models based on model optimized prediction method. *Neurocomputing*, *174*, 1087–1100. https://doi.org/10.1016/j.neucom.2015.09.105

Chen, S., Ho, D. W. C., & Huang, C. (2015). Fault Reconstruction and State Estimator Design for Distributed Sensor Networks in Multitarget Tracking. *IEEE Transactions on Industrial Electronics*, *62*(11), 7091–7102. https://doi.org/10.1109/TIE.2015.2448685

Chen, W. H., & You, F. (2020). Efficient Greenhouse Temperature Control with Data-Driven Robust Model Predictive. *Proceedings of the American Control Conference*, *2020-July*, 1986–1991. https://doi.org/10.23919/ACC45564.2020.9147701

Choomkasien, P., Chomphooyod, P., & Banjerdpongchai, D. (2017). Design of model predictive control for industrial process with input time delay. *International Conference on Control, Automation and Systems*, *2017-Octob*(Iccas), 625–630. https://doi.org/10.23919/ICCAS.2017.8204305

Cook, F. (2020). *Development of Apparatus for Ice Nucleation Studies*. http://research-information.bristol.ac.uk

Cooper, D. J. (2005). *Practical Process Control ® by Control Control Control Station ® Station Station ® Innovative Solutions from the Process Control Professionals Practical Process Control using LOOP-PRO Software*. www.controlstation.com

Dargie, W., & Poellabauer, C. (2011). Fundamentals of Wireless Sensor Networks: Theory and Practice. In *Fundamentals of Wireless Sensor Networks: Theory and Practice*. A John Wiley and Sons, Ltd. https://doi.org/10.1002/9780470666388

Das, A., Rakshit, A., & Parewa, H. P. (2011). Unrealized Potential of Sensor in Precision Agriculture. *International Journal of Agriculture, Environment and Biotechnology*, *4*(1), 99–102.

De Marcellis, A., & Ferri, G. (2011). *Analog Circuits and Systems for Voltage-Mode and Current-Mode Sensor Interfacing Applications*. https://doi.org/10.1007/978-90-481-9828-3

Ding, Y., Wang, L., Li, Y., & Li, D. (2018). Model predictive control and its application in agriculture: A review. *Computers and Electronics in Agriculture*, *151*(May), 104–117. https://doi.org/10.1016/j.compag.2018.06.004

El Ghoumari, M. Y., Tantau, H. J., & Serrano, J. (2005). Non-linear constrained MPC: Real-time implementation of greenhouse air temperature control. *Computers and Electronics in Agriculture*, *49*(3), 345–356. https://doi.org/10.1016/j.compag.2005.08.005

Elhadi, S., Marzak, A., & Sael, N. (2020). Operating models of Network protocols IoT: Short-range protocols. *2020 International Symposium on Advanced Electrical and Communication Technologies, ISAECT 2020*. https://doi.org/10.1109/ISAECT50560.2020.9523646

Er. Pooja Yadav, & Er. Ankur Mittal. (2016). A survey of growth and opportunity of Internet of Things (IoT) in Global Scenario. *Ijircce*, *4*(12).

Ferentinos, K. P., Katsoulas, N., Tzounis, A., Bartzanas, T., & Kittas, C. (2016). ScienceDirect Wireless sensor networks for greenhouse climate and plant condition assessment. *Biosystems Engineering*, *153*, 70–81. https://doi.org/10.1016/j.biosystemseng.2016.11.005

Ferreira, P. M., & Ruano, A. E. (2008). Discrete Model-Based Greenhouse Environmental Control using the Branch & Bound Algorithm. In *IFAC Proceedings Volumes* (Vol. 41, Issue 2). IFAC. https://doi.org/10.3182/20080706-5-kr-1001.00494

Fitz-Rodríguez, E., Kubota, C., Giacomelli, G. A., Tignor, M. E., Wilson, S. B., & McMahon, M. (2010). Dynamic modeling and simulation of greenhouse environments under several scenarios: A web-based application. *Computers and Electronics in Agriculture*, *70*(1), 105–116. https://doi.org/10.1016/j.compag.2009.09.010

Georgieva, T., Paskova, N., Gaazi, B., Todorov, G., & Daskalov, P. (2016). Design of Wireless Sensor Network for Monitoring of Soil Quality Parameters. *Agriculture and Agricultural Science Procedia*, *10*, 431–437. https://doi.org/10.1016/j.aaspro.2016.09.011

Goap, A., Sharma, D., Shukla, A. K., & Krishna, C. R. (2018). An IoT based smart irrigation management system using Machine learning and open source technologies. *Computers and Electronics in Agriculture*, *155*(October), 41–49. https://doi.org/10.1016/j.compag.2018.09.040

Guo, Y., Zhao, H., Zhang, S., Wang, Y., & Chow, D. (2021). Modeling and optimization of environment in agricultural greenhouses for improving cleaner and sustainable crop production. *Journal of Cleaner Production*, *285*, 124843. https://doi.org/10.1016/j.jclepro.2020.124843

GUTMAN, P. O., LINDBERG, I. I., & SEGINER, I. (n.d.). *A non-linear optimal greenhouse control problem solved by linear programming*.

Hamad, I. H., Chouchaine, A., & Bouzaouache, H. (2021). On Modeling Greenhouse Air-Temperature: An Experimental Validation. *18th IEEE International Multi-Conference on Systems, Signals and Devices, SSD 2021*, 353–358. https://doi.org/10.1109/SSD52085.2021.9429311

Haugen, F. A. (2021). *Modeling, Simulation and Control*. *August*. http://techteach.no/control/modsimcon/mod_sim_con_2021_08_20_v01.pdf

Haugen TechTeach, F. (2010). *Basic DYNAMICS and CONTROL*.

Henten, V. E. J. (1994). *Greenhouse Climate management: An optimal control approach*.

Horatiu, G. E., & Andreescu, G.-D. (2012). *Comparison study of PID controller tuning for greenhouse climate with feedback-feedforward linearization and decoupling*. https://www.researchgate.net/publication/260479438

Hu, H., Xu, L., Goodman, E. D., & Zeng, S. (2014). NSGA-II-based nonlinear PID controller tuning of greenhouse climate for reducing costs and improving performances. *Neural Computing and Applications*, *24*(3–4), 927–936. https://doi.org/10.1007/s00521-012-1312-8

Hu, H., Xu, L., Wei, R., & Zhu, B. (2011). Multi-objective control optimization for greenhouse environment using evolutionary algorithms. *Sensors*, *11*(6), 5792–5807. https://doi.org/10.3390/s110605792

HWANG, Y. (1993). *Optimization of Greenhouse Temperature*.

Iddio, E., Wang, L., Thomas, Y., McMorrow, G., & Denzer, A. (2020a). Energy efficient operation and modeling for greenhouses: A literature review. *Renewable and Sustainable Energy Reviews*, *117*(November 2019), 109480. https://doi.org/10.1016/j.rser.2019.109480

Iddio, E., Wang, L., Thomas, Y., McMorrow, G., & Denzer, A. (2020b). Energy efficient operation and modeling for greenhouses: A literature review. *Renewable and Sustainable Energy Reviews*, *117*(November 2019), 109480. https://doi.org/10.1016/j.rser.2019.109480

Jain, D., & Tiwari, G. N. (2003). Modeling and optimal design of ground air collector for heating in controlled environment greenhouse. *Energy Conversion and Management*, *44*(8), 1357–1372. https://doi.org/10.1016/S0196-8904(02)00118-8

Jan, B., Farman, H., Javed, H., Montrucchio, B., Khan, M., & Ali, S. (2017). Energy efficient in-network aggregation algorithms in wireless sensor

networks: A survey. *Wireless Communications and Mobile Computing*, *397*, 135–148. https://doi.org/10.1007/978-981-10-1627-1_11

Jiang, J., & Moallem, M. (2020). Development of an Intelligent LED Lighting Control Testbed for IoT-based Smart Greenhouses. *IECON Proceedings (Industrial Electronics Conference)*, *2020-Octob*, 5226–5231. https://doi.org/10.1109/IECON43393.2020.9254993

Jiang, X. (2020). *IoT Sensors For Industrial And Agricultural Applications: Development Of Wireless Network And Process Control*. *December*, 1–111.

Joudi, K. A., & Farhan, A. A. (2015). A dynamic model and an experimental study for the internal air and soil temperatures in an innovative greenhouse. *Energy Conversion and Management*, *91*, 76–82. https://doi.org/10.1016/j.enconman.2014.11.052

Kalaivani, T., Allirani, A., & Priya, P. (2011). A survey on Zigbee based wireless sensor networks in agriculture. *TISC 2011 - Proceedings of the 3rd International Conference on Trendz in Information Sciences and Computing*, *i*, 85–89. https://doi.org/10.1109/TISC.2011.6169090

Katzin, D., van Henten, E. J., & van Mourik, S. (2022a). Process-based greenhouse climate models: Genealogy, current status, and future directions. *Agricultural Systems*, *198*(January), 103388. https://doi.org/10.1016/j.agsy.2022.103388

Katzin, D., van Henten, E. J., & van Mourik, S. (2022b). Process-based greenhouse climate models: Genealogy, current status, and future directions. *Agricultural Systems*, *198*(January), 103388. https://doi.org/10.1016/j.agsy.2022.103388

Kevin, Asthon. (2010). That ’ Internet of Things ’ Thing. *RFiD Journal*, 4986. http://www.itrco.jp/libraries/RFIDjournal-That Internet of Things Thing.pdf

Khairunnniza-Bejo, S., Ramli, N. H., & Muharam, F. M. (2018). Wireless sensor network (WSN) applications in plantation canopy areas: A review. *Asian*

Journal of Scientific Research, *11*(2), 151–161. https://doi.org/10.3923/ajsr.2018.151.161

Lakhwani, K., Gianey, H., Agarwal, N., & Gupta, S. (2019). Development of IoT for Smart Agriculture a Review. *Emerging Trends in Expert Applications and Security, Advances*, *841*, 425–432. https://doi.org/10.1007/978-981-13-2285-3

Lautenschlager, B. (2019a). *Data-Driven Learning and Model Predictive Control for Heating Systems*.

Lautenschlager, B. (2019b). *Data-Driven Learning and Model Predictive Control for Heating Systems*.

Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., Hill, J., Welsh, M., Brewer, E., & Culler, D. (2005). TinyOS: An operating system for sensor networks. *Ambient Intelligence*, 115–148. https://doi.org/10.1007/3-540-27139-2_7

Li, H., Guo, Y., Zhao, H., Wang, Y., & Chow, D. (2021). Towards automated greenhouse: A state of the art review on greenhouse monitoring methods and technologies based on internet of things. *Computers and Electronics in Agriculture*, *191*(May), 106558. https://doi.org/10.1016/j.compag.2021.106558

Li, Y., Ding, Y., Li, D., & Miao, Z. (2018). Automatic carbon dioxide enrichment strategies in the greenhouse: A review. *Biosystems Engineering*, *171*, 101–119. https://doi.org/10.1016/j.biosystemseng.2018.04.018

Liang, M. H., Chen, L. J., He, Y. F., & Du, S. F. (2018). Greenhouse temperature predictive control for energy saving using switch actuators. *IFAC-PapersOnLine*, *51*(17), 747–751. https://doi.org/10.1016/j.ifacol.2018.08.106

Lijun, C., Shangfeng, D., Meihui, L., & Yaofeng, H. (2018). Adaptive Feedback Linearization-based Predictive Control for Greenhouse Temperature. *IFAC-*

PapersOnLine, *51*(17), 784–789. https://doi.org/10.1016/j.ifacol.2018.08.100

Liu, X., Gao, H., Sun, Y., Wu, Y., Martin, B., Chilton, J., Mirzaei, P., Zhang, X., Beccarelli, P., & Lau, B. (2016). Thermal and Optical Analysis of a Passive Heat Recovery and Storage System for Greenhouse Skin. *Procedia Engineering*, *155*(0), 472–478. https://doi.org/10.1016/j.proeng.2016.08.050

Lokke, M. M., Seefeldt, H. F., Edwards, G., & Green, O. (2011). Novel wireless sensor system for monitoring oxygen, temperature and respiration rate of horticultural crops post harvest. *Sensors*, *11*(9), 8456–8468. https://doi.org/10.3390/s110908456

Lvova, L., & Nadporozhskaya, M. (2017). Chemical sensors for soil analysis: principles and applications. In *New Pesticides and Soil Sensors* (pp. 637–678). https://doi.org/10.1016/b978-0-12-804299-1.00018-7

Maciejowski, J. W. (2002). *Predictive control with constraints*. www.peersoneduc.com

Mahdavian, M., & Wattanapongsakorn, N. (2014). Optimizing PID controller tuning for greenhouse lighting control system by varying number of objectives. *2014 11th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, ECTI-CON 2014*, *1*, 0–5. https://doi.org/10.1109/ECTICon.2014.6839890

Maher, A., Kamel, E., Enrico, F., Atif, I., & Abdelkader, M. (2016). An intelligent system for the climate control and energy savings in agricultural greenhouses. *Energy Efficiency*, *9*(6), 1241–1255. https://doi.org/10.1007/s12053-015-9421-8

Manonmani, A., Thyagarajan, T., Sutha, S., & Gayathri, V. (2016). Design of Soft Computing Based Optimal PI Controller for Greenhouse System. *Circuits and Systems*, *07*(11), 3431–3447. https://doi.org/10.4236/cs.2016.711292

Markovic, D., Koprivica, R., Pesovic, U., & Randic, S. (2016). Application of IoT in monitoring and controlling agricultural production. *Acta Agriculturae Serbica*, *20*(40), 145–153. https://doi.org/10.5937/aaser1540145m

Mayne, D. Q. (2014). Model predictive control: Recent developments and future promise. *Automatica*, *50*(12), 2967–2986. https://doi.org/10.1016/j.automatica.2014.10.128

McNulty, J. (2017). Solar greenhouses generate electricity and grow crops at the same time, UC Santa Cruz study reveals. *UC Santa Cruz Magazine*, *November*, 1–2. https://doi.org/10.7291/D10T0W

Messai, M. L., & Seba, H. (2016). A survey of key management schemes in multi-phase wireless sensor networks. *Computer Networks*, *105*, 60–74. https://doi.org/10.1016/j.comnet.2016.05.005

Mogal, P. W., & Warke, N. (2013). Model Predictive Control using LabVIEW. *2013 International Conference on Advances in Technology and Engineering, ICATE 2013*. https://doi.org/10.1109/ICAdTE.2013.6524719

Montoya-Ríos, A. P., García-Mañas, F., Guzmán, J. L., & Rodríguez, F. (2020). Simple tuning rules for feedforward compensators applied to greenhouse daytime temperature control using natural ventilation. *Agronomy*, *10*(9). https://doi.org/10.3390/agronomy10091327

Navarro-Hellín, H., Martínez-del-Rincon, J., Domingo-Miguel, R., Soto-Valles, F., & Torres-Sánchez, R. (2016). A decision support system for managing irrigation in agriculture. *Computers and Electronics in Agriculture*, *124*, 121–131. https://doi.org/10.1016/j.compag.2016.04.003

Ojha, T., Misra, S., & Raghuwanshi, N. S. (2015). Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges. *Computers and Electronics in Agriculture*, *118*, 66–84. https://doi.org/10.1016/j.compag.2015.08.011

Pajares, G., Peruzzi, A., & Gonzalez-de-Santos, P. (2013). Sensors in agriculture and forestry. *Sensors (Switzerland)*, *13*(9), 12132–12139. https://doi.org/10.3390/s130912132

Patil, R., & Shruti. (2016). An Approach for Agricultural Field Monitoring and Control Using IoT. *International Research Journal of Engineering and Technology*, *3*(9), 86–89.

Peng, C., & Han, Q. L. (2016). On Designing a Novel Self-Triggered Sampling Scheme for Networked Control Systems with Data Losses and Communication Delays. *IEEE Transactions on Industrial Electronics*, *63*(2), 1239–1248. https://doi.org/10.1109/TIE.2015.2504044

Ramakrishnan, R., Sasikala, K., & Nagappan, A. (2020). Issue 11 www.jetir.org (ISSN-2349-5162). In *JETIR2011065 Journal of Emerging Technologies and Innovative Research* (Vol. 7). www.jetir.org

Rice, R. C. (2010). *PID Tuning Guide A Best-Practices Approach to Understanding and Tuning PID Controllers First Edition Technical Contributions from: Also Introducing: Simplifying PID Control, Optimizing Plant Performance*.

Roberts, P. D. (1995). *the sequence calculated at each step. The strategy is illustrated as shown in Figure 1 and described as follows (Camacho and Bordons, 1995):-*. 1–4.

Rodríguez, S., Gualotuña, T., & Grilo, C. (2017). A System for the Monitoring and Predicting of Data in Precision Agriculture in a Rose Greenhouse Based on Wireless Sensor Networks. *Procedia Computer Science*, *121*, 306–313. https://doi.org/10.1016/j.procs.2017.11.042

Romero, R., Muriel, J. L., García, I., & Muñoz de la Peña, D. (2012). Research on automatic irrigation control: State of the art and recent results. *Agricultural Water Management*, *114*, 59–66. https://doi.org/10.1016/j.agwat.2012.06.026

Roy, J. C., Boulard, T., Kittas, C., & Wang, S. (2002). Convective and ventilation transfers in greenhouses, part 1: The greenhouse considered as a perfectly stirred tank. *Biosystems Engineering*, *83*(1), 1–20. https://doi.org/10.1006/bioe.2002.0107

Ruiz-Garcia, L., Lunadei, L., Barreiro, P., & Robla, J. I. (2009). A review of wireless sensor technologies and applications in agriculture and food industry: State of the art and current trends. *Sensors (Switzerland)*, *9*(6), 4728–4750. https://doi.org/10.3390/s90604728

Sagheer, A., Mohammed, M., Riad, K., & Alhajhoj, M. (2021). A cloud-based IoT platform for precision control of soilless greenhouse cultivation. *Sensors (Switzerland)*, *21*(1), 1–29. https://doi.org/10.3390/s21010223

Salvi, S., Jain, P. S. A., A, H. J., K, T. H., Farhana, M., Jain, N., & V, M. S. (2017). Cloud Based Data Analysis and Monitoring of Smart Multi-level Irrigation System Using IoT. *International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC 2017) Cloud*, 1292–1295.

Sangwan, A., Gupta, A., Tiwari, D., Kumar, V., & Rana, K. P. S. (2019). Comparative Study of Optimization Techniques for Tuning of PI Gains for Greenhouse Climate Control. *2019 International Conference on Computing, Power and Communication Technologies, GUCON 2019*, 274–280.

Šaš, M., & Pejić, D. (2021). The application of IoT solutions in greenhouses with the aim of reducing electrical energy consumption. *Journal on Processing and Energy in Agriculture*, *25*(4), 156–159. https://doi.org/10.5937/jpea25-34848

Serale, G., Gnoli, L., Giraudo, E., & Fabrizio, E. (2021). A supervisory control strategy for improving energy efficiency of artificial lighting systems in greenhouses. *Energies*, *14*(2), 1–19. https://doi.org/10.3390/en14010202

Sethi, V. P., Sumathy, K., Lee, C., & Pal, D. S. (2013). Thermal modeling aspects of solar greenhouse microclimate control: A review on heating technologies. *Solar Energy*, *96*, 56–82. https://doi.org/10.1016/j.solener.2013.06.034

Shamim, A. (2018). *Thermal environment modeling and optimization of greenhouse in cold regions* (Issue April).

Shen, Y., Wei, R., & Xu, L. (2018). Energy consumption prediction of a greenhouse and optimization of daily average temperature. *Energies*, *11*(1). https://doi.org/10.3390/en11010065

Sigrimis, N., Arvanitis, K., Kookos, I. K., & Paraskevopoulos, P. N. (1999). *H-PI CONTROLLERTUNINGFORGREENHOUSETEMPERATURECONTROL*. Pergamon.

Sinclair, I. R. (2001). *Sensors and Transducers Third edition* (Third edit). http://senofficial.yolasite.com/resources/sensors and transducers.pdf

Singh, M. C., Singh, J. P., & Singh, K. G. (2018). Development of a microclimate model for prediction of temperatures inside a naturally ventilated greenhouse under cucumber crop in soilless media. *Computers and Electronics in Agriculture*, *154*(August), 227–238. https://doi.org/10.1016/j.compag.2018.08.044

Soribe, F. I., & Curry, R. B. (1973). Simulation of lettuce growth in an air-supported plastic greenhouse. *Journal of Agricultural Engineering Research*, *18*(2), 133–140. https://doi.org/10.1016/0021-8634(73)90022-X

Speetjens, S. L., Stigter, J. D., & van Straten, G. (2010). Physics-based model for a water-saving greenhouse. *Biosystems Engineering*, *105*(2), 149–159. https://doi.org/10.1016/j.biosystemseng.2009.06.026

Sreekantha, D. K. (2016). Automation in Agriculture : a Study. *International Journal of Engineering Science Invention Research & Development*, *2*(7), 822–833. https://pdfs.semanticscholar.org/0b11/ef8232e16c5b67a30643880db1d7dbe57baa.pdf

STANGHELLINI, C. (n.d.). *Transpiration of greenhouse crops an Aid To Climate Management*.

Su, Y., Yu, Q., & Zeng, L. (2020). Parameter self-tuning pid control for greenhouse climate control problem. *IEEE Access*, *8*, 186157–186171. https://doi.org/10.1109/ACCESS.2020.3030416

Thakur, D., Kumar, Y., Kumar, A., Kumar, P., & Singh, V. (2018). ScienceDirect Real Time Monitoring of Valeriana Jatamansi Plant for Growth Analysis. *Procedia Computer Science*, *132*, 507–517. https://doi.org/10.1016/j.procs.2018.05.003

Thakur, D., Kumar, Y., Kumar, A., & Singh, P. K. (2019). Applicability of Wireless Sensor Networks in Precision Agriculture: A Review. In *Wireless Personal Communications* (Issue 0123456789). Springer US. https://doi.org/10.1007/s11277-019-06285-2

Thomopoulos, V., Bitas, D., Papastavros, K. N., Tsipianitis, D., & Kavga, A. (2021). Development of an integrated IoT-based greenhouse control three-device robotic system. *Agronomy*, *11*(2), 1–16. https://doi.org/10.3390/agronomy11020405

Tiwari, G. N., Sharma, P. K., Goyal, R. K., & Sutar, R. F. (1998). Estimation of an efficiency factor for a greenhouse: A numerical and experimental study. *Energy and Buildings*, *28*(3), 241–250. https://doi.org/10.1016/s0378-7788(97)00062-5

Tunc, M., Venart, J. E. S., & Sollows, K. (1985). Bivalent (hybrid) heat pump-oil heating systems for greenhouses. *Journal of Heat Recovery Systems*, *5*(6), 483–491. https://doi.org/10.1016/0198-7593(85)90215-2

Udink ten Cate, A. J. (1984). Modelling and (adaptive) control of greenhouse climates. *Netherlands Journal of Agricultural Science*, *32*(2), 137–139. https://doi.org/10.18174/njas.v32i2.16912

Ulpiani, G., Borgognoni, M., Romagnoli, A., & Di Perna, C. (2016). Comparing the performance of on/off, PID and fuzzy controllers applied to the heating system of an energy-efficient building. *Energy and Buildings*, *116*(November 2017), 1–17. https://doi.org/10.1016/j.enbuild.2015.12.027

Vadiee, A. (2013). *Energy Management in Large scale Solar Buildings : The Closed Greenhouse Concept*.

Van Beveren, P. J. M., Bontsema, J., Van Straten, G., & Van Henten, E. J. (2013). Minimal heating and cooling in a modern rose greenhouse. In *IFAC Proceedings Volumes (IFAC-PapersOnline)* (Vol. 4, Issue PART 1). IFAC. https://doi.org/10.3182/20130828-2-SF-3019.00026

van Beveren, P. J. M., Bontsema, J., van Straten, G., & van Henten, E. J. (2015). Optimal control of greenhouse climate using minimal energy and grower defined bounds. *Applied Energy*, *159*(2015), 509–519. https://doi.org/10.1016/j.apenergy.2015.09.012

Van Ooteghem, R. J. C., Van Willigenburg, L. G., & Van Straten, G. (2005). Receding horizon optimal control of a solar greenhouse. *Acta Horticulturae*, *691*(January 2016), 797–806. https://doi.org/10.17660/ActaHortic.2005.691.98

Varghese, R., & Sharma, S. (2019). Affordable Smart Farming Using IoT and Machine Learning. *Proceedings of the 2nd International Conference on Intelligent Computing and Control Systems, ICICCS 2018*, *Iciccs*, 645–650. https://doi.org/10.1109/ICCONS.2018.8663044

Vijverberg, A. J., & Strijbosch, T. (1968). *Ontwikkelingen in de klimaatregeling bij tomaat en sla Ontwikkelingen in de klimaatregeling bij tomaat en sla*. *31*.

Villaverde, B. C., Rea, S., & Pesch, D. (2012). InRout - A QoS aware route selection algorithm for industrial wireless sensor networks. *Ad Hoc Networks*, *10*(3), 458–478. https://doi.org/10.1016/j.adhoc.2011.07.015

Vineela, T., NagaHarini, J., Kiranmai, C., Harshitha, G., & AdiLakshmi, B. (2018). IoT Based Agriculture Monitoring and Smart Irrigation System Using Raspeberry Pi. *International Research Journal of Engineering and Technology*, *5*(1), 1417–1420. https://www.irjet.net/archives/V5/i1/IRJET-V5I1307.pdf

Vukov, M., Gros, S., Horn, G., Frison, G., Geebelen, K., Jørgensen, J. B., Swevers, J., & Diehl, M. (2015). Real-time nonlinear MPC and MHE for a large-scale mechatronic application. *Control Engineering Practice*, *45*, 64–78. https://doi.org/10.1016/j.conengprac.2015.08.012

Wang, J., Wang, H., He, J., Li, L., Shen, M., Tan, X., Min, H., & Zheng, L. (2015). Wireless sensor network for real-time perishable food supply chain management. *COMPUTERS AND ELECTRONICS IN AGRICULTURE*, *110*, 196–207. https://doi.org/10.1016/j.compag.2014.11.009

Wang, Q., & Balasingham, I. (2010). Wireless Sensor Networks - An Introduction. In *Wireless Sensor Networks: Application-Centric Design*. Web of Science. https://doi.org/10.5772/13225

Wang, Y. (2019). *The IoT Based Environmental Sensing Platform by*. *June*.

Watt, A. J., Phillips, M. R., Campbell, C. E., Wells, I., & Hole, S. (2019). Science of the Total Environment Wireless Sensor Networks for monitoring underwater sediment transport. *Science of the Total Environment*, *667*, 160–165. https://doi.org/10.1016/j.scitotenv.2019.02.369

Xu, J., Gu, B., & Tian, G. (2022). Review of agricultural IoT technology. *Artificial Intelligence in Agriculture*, *6*, 10–22. https://doi.org/10.1016/j.aiia.2022.01.001

Yang, X., Zhang, W. A., Chen, M. Z. Q., & Yu, L. (2017). Hybrid sequential fusion estimation for asynchronous sensor network-based target tracking. *IEEE Transactions on Control Systems Technology*, *25*(2), 669–676. https://doi.org/10.1109/TCST.2016.2558632

Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks*, *52*(12), 2292–2330. https://doi.org/10.1016/j.comnet.2008.04.002

Zamora-Izquierdo, M. A., Santa, J., Martínez, J. A., Martínez, V., & Skarmeta, A. F. (2019). Smart farming IoT platform based on edge and cloud computing.

Biosystems Engineering, *177*, 4–17. https://doi.org/10.1016/j.biosystemseng.2018.10.014

Zhang, J., & Varadharajan, V. (2010). Wireless sensor network key management survey and taxonomy. *Journal of Network and Computer Applications*, *33*(2), 63–75. https://doi.org/10.1016/j.jnca.2009.10.001

Zhang, S., Guo, Y., Zhao, H., Wang, Y., Chow, D., & Fang, Y. (2020a). Methodologies of control strategies for improving energy efficiency in agricultural greenhouses. *Journal of Cleaner Production*, *274*, 122695. https://doi.org/10.1016/j.jclepro.2020.122695

Zhang, S., Guo, Y., Zhao, H., Wang, Y., Chow, D., & Fang, Y. (2020b). Methodologies of control strategies for improving energy efficiency in agricultural greenhouses. *Journal of Cleaner Production*, *274*. https://doi.org/10.1016/j.jclepro.2020.122695

Zou, Q., Ji, A., Zhang, S., Shi, M., & Luo, Y. (2010). Model predictive control based on particle swarm optimization of greenhouse climate for saving energy consumption. *2010 World Automation Congress, WAC 2010*, 123–128.

Appendices:

List of publications:

1. Faniyi, B, and Luo, Z. A Physics-Based Modelling and Control of Greenhouse System Air Temperature Aided by IoT Technology. Published in *Energies*.
2. A comparison of On-Off, PI, PID and MPC control implementations in greenhouse temperature control and energy minimization. Manuscript in preparation for submission to *IEEE Access*.

Appendix 3.1

Greenhouse simulation

```
function f = TinFun_ODE(A,B)
Tin=B(1);                    %Parameters and explicit Equations
Crad = 0.1;                  % transmission of roof
Qrad = 72;                   % solar irradiations from outside weather
Ag = 12.5;                   % greenhouse surface area
Qcoe = 0.21;                 % polycarbonate convective heat transfer coefficient
Tout = 23;                   % gotten from the Cranfield environmental dataset
Ac = 2.2;                    % greenhouse cover material area
Ts = 20;                     % soil surface temperature
Rhoa = 1.29;                 % air density
Ca = 1000;
Kcoe = -7.88;                % soil heat transfer coefficient
k = 1.3269;
TitaW = 1.0857;
v = 0.4444;
w = 3.4;                     % external wind speed
Uw = 0;                      % window opening assumed closed (initial)
Aw = 0.36;                   % roof window total area
alpha = 60;                  % window opening angle
Uh3 = 1;                     % heater opening
Kcoe3 = -7.88;
Tp3 = 250;
Uh1 = 1;
Kcoe1 = -7.88;
Tp1 = 50;
Uh2 = 1;
Kcoe2 = -7.88;
Tp2 = 25;
```

```
Uh4 = 1;
Kcoe4 = -7.88;
Tp4 = 50;
N = 5;                          % 5 used as part of electric energy transformed to heat
Ep = 100;
Fon = 100;
Vg = 3.87;
Qsun = Crad * Qrad * Ag;
Qcov = Qcoe * (Tout - Tin) * Ac;
Qsoil = Kcoe * (Ts - Tin) * Ag;
Nven = k + TitaW + (v * w * Uw);
Aven = 22 * Aw * sin(alpha/2);
Qvent = Rhoa * Ca * Nven * (Tout - Tin) * Aven;
Qheater3 = Uh3 * Kcoe3 * (Tp3 - Ts);
Qheater1 = Uh1 * Kcoe1 * (Tp1 - Ts);
Qheater2 = Uh2 * Kcoe2 * (Tp2 - Ts);
Qheater4 = Uh4 * Kcoe4 * (Tp4 - Ts);
Qenergy = 0 - Qvent + Qheater1 + Qheater2 + Qheater3 + Qheater4;
Qlamp = N * Ep * (Fon/100);
Ccap = Rhoa * Ca * Vg;          % Differential equation
dTindt = 1/Ccap * (Qsun - Qcov - Qsoil + Qlamp + Qenergy);
f = [dTindt];
end

clc
Tspan = [0 24];                 %range for the independent varaible i.e. time
a0 = [0];                       % Initial value for the dependent varaible Tin
[a b]=ode45(@TinFun_ODE,Tspan,a0);
plot(a,b(:,1));
legend('Tin')
ylabel('Indoor temp(°C)');
xlabel('time(s)');
axis([0 24 0 30]);              % this is the x and yaxis scale
title('Greenhouse indoor temperature')
```

Appendix 3.2

```
# Prelimenary work - to estimate two parameters to fit data
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint
from scipy.optimize import minimize
import pandas as pd
import math

data = pd.read_csv("C:\\Users\\user\\OneDrive\\Desktop\\Environ data\\Excel-to-text3.csv")
```

```
t = data['time'].values - data['time'].values[0]
u = data['u'].values
Tinp = data['y'].values
Tinp0 = Tinp[0]
u0 = u[0]

ns = len(t)
delta_t = t[1]-t[0]                    #time step for each time interval

# define energy balance model
def greenhouse(x,t,u,p):
    # Optimized parameters
    Ts,Qven = p
# %% Model parameters
    Crad = 0.1;                        # transmission of roof
    Ag = 12.5                          # greenhouse surface areate
    Ac = 2.2                           # greenhouse cover material area
    Rhoa = 1.29                        # air density
    Ca = 1000.0
    Vg = 3.87
    Qcoe = 0.21
    Kcoe = -7.88                       # soil heat transfer coefficient
# Temperature States
    Tin_k = x[0]
    Qrad_k = 108.2
    Tout_k = 10.5
# Derived equations
    Qsun = Crad * Qrad_k *Ag;
    Qcov = Qcoe * (Tout_k - Tin_k) * Ac;
    Qsoil = Kcoe * (Ts - Tin_k) * Ag;
    Ccap = Rhoa * Ca * Vg;
    Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k);
# calculate derivative
    dTindt = (1/Ccap) * (Qsun - Qcov - Qsoil - Qvent + u)

    return [dTindt]

def simulate(p):
    # input argumentss
    Ts = p[0]
    Qven = p[1]
# storage for model values
    Tinm = np.zeros(ns)
# initial condition
    Tinm[0] = Tinp0
# loop through time steps
    for i in range(len(t)-1):
        ts = [t[i],t[i+1]]
```

```
        y = odeint(greenhouse,Tinm[i],ts,args=(u[i],p))
        Tinm[i+1] = y[-1]
    return Tinm
# define objective
def objective(p):
# simulate model
    Tinm = simulate(p)
# calculate objective
    obj = 0.0
    for i in range(len(Tinm)):
        obj = obj + (Tinm[i]-Tinp[i])**2
# return result
    return obj
# Parameter initial guess
p0 = np.zeros(2)
p0[0] = 20.58          # Heat transfer coefficient (W/m^2-K)
p0[1] = -5.88          # Heat gain 1 (W/%)
# show initial objective
print('Initial SSE Objective: ' + str(objective(p0)))
solution = minimize(objective,p0)
p = solution.x
# show final objective
print('Final SSE Objective: ' + str(objective(p)))
# optimized parameter values
print('Ts: ' + str(p[0]))
print('Qven: ' + str(p[1]))
# calculate model with updated parameters
Tinm1  = simulate(p0)
Tinm2  = simulate(p)
# Plot results
plt.figure(1)
# Plot results
plt.figure(1)
plt.close('all')
plt.figure(figsize=(10,7))
plt.subplot(2,1,1)
plt.grid()
plt.plot(t,Tinp,'g-',linewidth=1,label='TinMeasured')
plt.plot(t,Tinm1,'b-',linewidth=1,label='Initial Guess')
plt.plot(t,Tinm2,'r--',linewidth=1,label='Optimized TinPredicted')
plt.xlabel('t [m]')
plt.ylim(16,40)
plt.ylabel('Output')
plt.legend(loc='best')
```

Appendix 3.3

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit, minimize
from sklearn.metrics import mean_absolute_error, r2_score
import pandas as pd

# Define FOPDT transfer function
def fopdt(t, Kp, tau_p, theta_p):
    y = Kp*(1-np.exp(-(t-theta_p)/tau_p))
    return y
# Import data file
data = pd.read_csv("C:\\Users\\user\\OneDrive\\Desktop\\Environ
data\\heater_data1.csv")
t = data['time'].values - data['time'].values[0]
u = data['u'].values
Tinp = data['y'].values
Tinp0 = Tinp[0]
u0 = u[0]
# Estimate FOPDT model parameters
p0 = [1.0, 2.0, 4.0] # initial guesses for Kp, tau_p, and theta_p
popt, pcov = curve_fit(fopdt, t, Tinp-Tinp0, p0=p0)
# Extract FOPDT model parameters
Kp, tau_p, theta_p = popt

# Simulate FOPDT response
t_sim = np.linspace(0, t[-1], 1000)
u_sim = np.ones_like(t_sim) * u0
Tinp_sim = fopdt(t_sim, Kp, tau_p, theta_p) + Tinp0

# Interpolate simulated output onto same time points as measured output
Tinp_sim_interp = np.interp(t, t_sim, Tinp_sim)
# Plot experimental and simulated data
plt.plot(t, Tinp, 'b-', label='Experimental')
plt.plot(t, Tinp_sim_interp, 'r-', label='Simulated')
plt.xlabel('Time (s)')
plt.ylabel('Temperature (deg C)')
plt.legend()
plt.show()

# Compute RMSE
print("Estimated FOPDT model parameters:")
print("Kp =", Kp)
print("tau_p =", tau_p)
print("theta_p =", theta_p)
rmse = np.sqrt(np.mean((Tinp-Tinp_sim_interp)**2))
print("RMSE =", rmse)

# Calculate the mae and R-squared
```

```
mae = mean_absolute_error(Tinp, Tinp_sim_interp)
print('mean_absolute_error:', mae)

r2 = r2_score(Tinp, Tinp_sim_interp)
print('R-squared:', r2)
# Define function to solve for steady-state input
def ss_input(Kp, tau_p, theta_p, T_ss):
    return Kp*(1-np.exp(-(theta_p-T_ss)/tau_p))
# Find input for 23 degrees Celsius
T_ss = 23.0
u_ss = minimize(lambda u: abs(ss_input(Kp, tau_p, theta_p, T_ss)-u), u0).x[0]
```

Appendix 3.4

```
# Real static operation for the two Controller PI and On-OFF
import matplotlib.pyplot as plt
import numpy as np
import math

Crad = 0.1
Qrad = 108.2
Ag = 12.5
Qcoe = 0.21
Tout = 10.0
Ac = 2.2
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
Kcoe = -7.88
Qven = 0.29

Tin_min = 0
Tin_max = 100

#%% Calculation of power giving specified static temp :
Tin_static = 23
Qsun = Crad * Qrad * Ag;
Qcov = Qcoe * (Tout - Tin_static) * Ac;
Qsoil = Kcoe * (Ts - Tin_static) * Ag;
Qvent = Qven * Rhoa * Ca * (Tout - Tin_static);

# From model after tin is set to zero (compute static value ):
Qenergy = - (Qsun - Qcov - Qsoil -Qvent)

print ('Qenergy [ W ] =' , Qenergy)
#%% Sim time settings :
dt = 1 # [ s ]
t_start = 0 # [ s ]
```

```
t_stop = 6000 # [ s ]
N_sim = int (( t_stop - t_start )/ dt ) + 1

#%% Preallocation of arrays for storing :
t_array = np . zeros ( N_sim )
Tin_array = np . zeros ( N_sim )
Tout_array = np . zeros ( N_sim )
Qenergy_array = np . zeros ( N_sim )

#%% Sim loop :
Tin_k = Tin_init = 17.5
for k in range (0 , N_sim ):
# State limitation :
    Tin_k = np . clip ( Tin_k , Tin_min , Tin_max )
    t_k = k * dt
    Tout_k = Tout
    Qenergy_k = Qenergy

    t_array [ k ] = t_k
    Tin_array [ k ] = Tin_k
    Tout_array [ k ] = Tout_k
    Qenergy_array [ k ] = Qenergy_k
    # Time derivative :
Qsun = Crad * Qrad * Ag;
Qcov = Qcoe * (Tout - Tin_k) * Ac;
Qsoil = Kcoe * (Ts - Tin_k) * Ag;
Qvent = Qven * Rhoa * Ca * (Tout - Tin_k);
Ccap = Rhoa * Ca * Vg;

dTin_dt_k = (1/Ccap) * (Qsun - Qcov - Qsoil -Qvent + Qenergy)
Tin_kp1 = Tin_k + dt * dTin_dt_k
# Time index shift :
Tin_k = Tin_kp1

# %% Plotting :
plt . close ('all')
plt . figure (1)
plt . subplot (2 , 1 , 1)
plt . plot ( t_array , Tin_array , 'r' , label = 'Tin')
plt . plot ( t_array , Tout_array , 'g' , label = 'Tout')
plt . legend ()
plt . grid ()
plt . xlabel ( 't [ s ]')
plt . ylabel ( '[ deg C ]')
plt . subplot (2 , 1 , 2)
plt . plot ( t_array , Qenergy_array , 'm' , label = 'PowerInput')
plt . legend ()
plt . grid ()
```

```
plt . xlabel ( 't [ s ]')
plt . ylabel ( '[ W ]')
plt . show ()
```

Appendix 4.1

```
# DOUBLET TEST
import numpy as np
import matplotlib.pyplot as plt
from scipy.integrate import odeint

# define Greenhouse model
def Greenhouse(x,t,u,Tout):
    Qenergy = u

    # States (1):
    Tink = x[0]

    # Parameters:
    Crad = 0.1
    Qrad = 108.2
    Ag = 12.5
    Qcoe = 0.21
    Ac = 2.2
    Ts = 11.05
    Rhoa = 1.29
    Ca = 1000.0
    Vg = 3.87
    Kcoe = -7.88
    Qven = 0.29

    Qsun = Crad * Qrad * Ag;
    Qcov = Qcoe * (Tout - Tink) * Ac;
    Qsoil = Kcoe * (Ts - Tink) * Ag;
    Qvent = Qven * Rhoa * Ca * (Tout - Tink);
    Ccap = Rhoa * Ca * Vg;

    # Calculate Temperature derivative
    dTinkdt = (1/Ccap) * (Qsun - Qcov - Qsoil - Qvent + Qenergy)

    # Return xdot:
    xdot = np.zeros(1)
    xdot[0] = dTinkdt
    return xdot

# Steady State Initial Conditions for the States
Tink_ss = 23.0

x0 = np.empty(1)
```

```
x0[0] = Tink_ss

# Steady State Initial Condition
u_ss = -3640.2
Tout = 10.5
# Time Interval (min)
t = np.linspace(0,25,251)

Tink = np.ones(len(t)) * Tink_ss
u = np.ones(len(t)) * u_ss

# Step greenhouse heater to 295
u[10:100] = -3650.2 #adding and subtracting 10
u[100:190] = -3630.2
u[190:] = -3640.2

# Simulate Greenhouse
for i in range(len(t)-1):
    ts = [t[i],t[i+1]]
    y = odeint(Greenhouse,x0,ts,args=(u[i+1],Tout))
    Tink[i+1] = y[-1][0]
    x0[0] = Tink[i+1]

data = np.vstack((t,u,Tink))
data = data.T
np.savetxt('data_doublet.txt1',data,delimiter=',',\
          header='Time,Qenergy,Tink',comments='')

# Plot the results
plt.figure(facecolor='None')
plt.figure(figsize=(6, 8))
plt.subplot(3,1,1)
plt.plot(t,u,'b--',lw=3)
plt.ylabel('Heating/Cooling input (W)')
plt.legend(['GH Heater'],loc='best')

plt.subplot(3,1,2)
plt.plot(t,Tink,'r-',lw=3)
plt.ylabel('Tin (C)')
plt.legend(['Indoor Temp'],loc='best')
plt.xlabel('t [m]')

# Add some space between subplots
plt.subplots_adjust(hspace=0.4)

plt.show()
```

Appendix 4. 2

```
# Def of On-off controller function:
import matplotlib.pyplot as plt
import numpy as np

# %% Def of on/off controller:
def fun_on_off(y_sp_k, y_k, controller_params,
               contr_state):
    # Control error:
    e_k = y_sp_k - y_k

    # Setting state of the controller:
    if e_k <= e_min:
        contr_state = False
    if e_k >= e_max:
        contr_state = True
    if (e_min <= e_k <= e_max) and (contr_state == True):
        contr_state = True
    if (e_min <= e_k <= e_max) and (contr_state== False):
        contr_state = False

    # Selecting control signal based on controller state:
    if contr_state == True:
        u_k = u_on
    if contr_state == False:
        u_k = u_off
    return (u_k, contr_state)

# %% Def of process model simulator
def process_sim(Tin_k, u_k, process_params, dt):
    (Qrad_k, Tout_k, Tin_min, Tin_max) = process_params
    Tin_k = np.clip(Tin_k, Tin_min, Tin_max)
    dTin_dt_k = (1/Ccap) * (Qsun - Qcov - Qsoil - Qvent + u_k)
    Tin_kp1 = Tin_k + dt*dTin_dt_k
    return Tin_kp1
# %% Model parameters:
Crad = 0.1;
Qrad_k = 108.2
Ag = 12.5
Qcoe = 0.21
Tout_k = 10.5
Ac = 2.2
Kcoe = 96.3
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
```

```
Kcoe = -7.88
Qven = 0.29
# %% Initialization:
Tin_init = 0  # [oC]
Tin_k = Tin_init  # [oC]
contr_state = True
# Derived parameters:
Qsun = Crad * Qrad_k *Ag;
Qcov = Qcoe * (Tout_k - Tin_k) * Ac;
Qsoil = Kcoe * (Ts - Tin_k) * Ag;
Ccap = Rhoa * Ca * Vg;
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k);

Tin_min = 0  # [oC] State limit
Tin_max = 50  # [oC] State limit
process_params = (Qrad_k, Tout_k, Tin_min, Tin_max)

# %% Simulation time settings:
dt = 0.1  # [s]
t_start = 0  # [s]
t_stop = 1440  # [s]
N_sim = int((t_stop - t_start)/dt) + 1

# %% Params of input signals:
Tout_k = 10.0
Tin_sp = 23.0

# %% Controller params:
u_on = 3640.2
u_off = -3640.2
e_max = 5  # [oC]
e_min = -5  # [oC]

controller_params = (u_on, u_off, e_max, e_min)

# %% Preallocation of arrays for plotting etc:
t_array = np.zeros(N_sim)
Tin_sp_array = np.zeros(N_sim)
Tin_array = np.zeros(N_sim)
Qrad_array = np.zeros(N_sim)
Tout_array = np.zeros(N_sim)
u_array = np.zeros(N_sim)

# %% State limits:

Tin_min = 0  # [oC] State limit
Tin_max = 50  # [oC] State limit
```

```
# %% Simulation loop:
for k in range(0, N_sim):
    t_k = k*dt
   # Setting input:

    if (t_k >= t_start):
        Tin_sp_k = Tin_sp

    # On/off controller:
    (u_k, contr_state) = fun_on_off(Tin_sp_k, Tin_k,
                                    controller_params,
                                    contr_state)

    # Process sim (Euler integration):
    Tin_kp1 = process_sim(Tin_k, u_k, process_params, dt)

    # Updating arrays for plotting:
    t_array[k] = t_k
    Tin_array[k] = Tin_k
    Tin_sp_array[k] = Tin_sp_k
    u_array[k] = u_k


    # Time shift:
    Tin_k = Tin_kp1

# %% Plotting:
plt.close('all')
plt.figure(num=1, figsize=(9, 7))
plt.suptitle('Greenhouse')
plt.close('all')

plt.subplot(2, 1, 1)
plt.plot(t_array, Tin_sp_array,'r', label='Tin_sp')
plt.plot(t_array, Tin_array,'b', label='Tin')
plt.legend()
plt.xlabel('t [s]')
plt.ylabel('[oC]')
plt.grid()

plt.subplot(2, 1, 2)
plt.plot(t_array, u_array, 'm', label='Power u = P')
plt.legend()
plt.grid()
plt.xlabel('t [m]')
plt.ylabel('[W]')
plt.show()
```

Appendix 4.3

```
# Def of Heuristics PI controller function:
import matplotlib.pyplot as plt
import numpy as np

def fun_pi(y_sp_k, y_k, u_i_km1, controller_params, dt):

    (Kp, Ti, u_man, u_min, u_max) = controller_params
    e_k = y_sp_k - y_k
    u_p_k = Kp*(y_sp_k - y_k)
    u_i_k = u_i_km1 + (Kp*dt/Ti)*e_k
    u_i_k = np.clip(u_i_k, -(u_max-1*u_man),(u_max-1*u_man))
    u_k = u_man + u_p_k + u_i_k                          # Total control signal
    u_k = np.clip(u_k, u_min, u_max)

    return (u_i_k, u_k)

# %% Def of process simulator:
def process_sim(Tin_k, u_k, process_params, dt):
    (Qrad_k, Tout_k, Tin_min, Tin_max) = process_params
    Tin_k = np.clip(Tin_k, Tin_min, Tin_max)
    dTin_dt_k = (1/Ccap) * (Qsun - Qcov - Qsoil - Qvent + u_k)
    Tin_kp1 = Tin_k + dt*dTin_dt_k                       # Euler integration

    return Tin_kp1

# %% Model parameters:
Crad = 0.1;
Qrad_k = 108.2
Ag = 12.5
Qcoe = 0.21
Tout_k = 10.5
Ac = 2.2
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
Kcoe = -7.88
Qven = 0.29

# %% Params of input signals:
Tout_k = 10.0
Tin_sp = 23.0
# %% Initialization:
Tin_init = 0  # [oC]
Tin_k = Tin_init  # [oC]
u_i_km1 = 0  # [W]
```

```
# Derived model params:
Qsun = Crad * Qrad_k *Ag;
Qcov = Qcoe * (Tout_k - Tin_k) * Ac;
Qsoil = Kcoe * (Ts - Tin_k) * Ag;
Ccap = Rhoa * Ca * Vg;
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k);

Tin_min = 0  # [oC] State limit
Tin_max = 50  # [oC] State limit

process_params = (Qrad_k, Tout_k, Tin_min, Tin_max)

# %% Simulation time settings:
dt = 0.1  # [s]
t_start = 0  # [s]
t_stop = 1440  # [s]
N_sim = int((t_stop - t_start)/dt) + 1  # Num time-steps

# %% PI controller settings:
Kp = 80.4 # W/K]
Ti = 5.9  # [s]

u_man = 0  # [W]
u_min = -3640.2 # [W]
u_max = 3640.2  # [W]

controller_params = (Kp, Ti, u_man, u_min, u_max)

# %% Preallocation of arrays for plotting etc:
t_array = np.zeros(N_sim)
Tin_sp_array = np.zeros(N_sim)
Tin_array = np.zeros(N_sim)
Qrad_array = np.zeros(N_sim)
Tout_array = np.zeros(N_sim)
u_array = np.zeros(N_sim)

# %% Simulation loop:
for k in range(0, N_sim):
    t_k = k*dt   # Time
    # Setting input:
    if (t_k >= 0):
        Tin_sp_k = Tin_sp
    # PI controller:
    (u_i_k, u_k) = fun_pi(Tin_sp_k, Tin_k, u_i_km1,
                          controller_params, dt)

    # Process sim (Euler integration):
```

```
    Tin_kp1 = process_sim(Tin_k, u_k, process_params, dt)
    # Updating arrays for plotting:
    t_array[k] = t_k
    Tin_array[k] = Tin_k
    Tin_sp_array[k] = Tin_sp_k
    Qrad_array[k] = Qrad_k
    Tout_array[k] = Tout_k
    u_array[k] = u_k

    # Time shift:
    Tin_k = Tin_kp1
    u_i_km1 = u_i_k

# %% Plotting:
plt.close('all')
plt.figure(1, figsize=(8, 5))
plt.subplot(2, 1, 1)
plt.plot(t_array, Tin_sp_array,'m', label='Tin_sp')
plt.plot(t_array, Tin_array,'b', label='Tin')
plt.legend()
plt.xlabel('t [s]')
plt.ylabel('[oC]')
plt.grid()

plt.subplot(2, 1, 2)
plt.plot(t_array, u_array, 'r', label='Power u = P')
plt.plot(t_array, Qrad_array,'g', label='SolarIrrad')
plt.plot(t_array, Tout_array,'y', label='OutsideTemp')
plt.legend()
plt.grid()
plt.xlabel('t [m]')
plt.ylabel('[W]')
plt.legend(loc=4)
plt.show()
```

Appendix 4. 4

```
# Def of Tuned PI controller function:
import matplotlib.pyplot as plt
import numpy as np

def fun_pi(y_sp_k, y_k, u_i_km1, controller_params, dt):

    (Kp, Ti, u_man, u_min, u_max) = controller_params
    e_k = y_sp_k - y_k
    u_p_k = Kp*(y_sp_k - y_k)
    u_i_k = u_i_km1 + (Kp*dt/Ti)*e_k
    u_i_k = np.clip(u_i_k, -(u_max-1*u_man),(u_max-1*u_man))
    u_k = u_man + u_p_k + u_i_k
```

```
    u_k = np.clip(u_k, u_min, u_max)

    return (u_i_k, u_k)

# %% Def of process simulator:

def process_sim(Tin_k, u_k, process_params, dt):

    (Qrad_k, Tout_k, Tin_min, Tin_max) = process_params
    Tin_k = np.clip(Tin_k, Tin_min, Tin_max)
    dTin_dt_k = (1/Ccap) * (Qsun - Qcov - Qsoil - Qvent + u_k)
    Tin_kp1 = Tin_k + dt*dTin_dt_k

    return Tin_kp1

# %% Model parameters:
Crad = 0.1;
Qrad_k = 108.2
Ag = 12.5
Qcoe = 0.21
Tout_k = 10.5
Ac = 2.2
Kcoe = 96.3
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
Kcoe = -7.88
Qven = 0.29

# %% Params of input signals:
Tout_k = 10.0
Tin_sp = 23.0

# %% Initialization:
Tin_init = 0  # [oC]
Tin_k = Tin_init  # [oC]
u_i_km1 = 0  # [W]

# Derived model params:
Qsun = Crad * Qrad_k *Ag;
Qcov = Qcoe * (Tout_k - Tin_k) * Ac;
Qsoil = Kcoe * (Ts - Tin_k) * Ag;
Ccap = Rhoa * Ca * Vg;
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k);

Tin_min = 0
Tin_max = 50
```

```
process_params = (Qrad_k, Tout_k, Tin_min, Tin_max)
# %% Simulation time settings:
dt = 0.1  # [s]
t_start = 0  # [s]
t_stop = 1440  # [s]
N_sim = int((t_stop - t_start)/dt) + 1  # Num time-steps
# %% PI controller settings:
Kp = 132.4 # W/K]
Ti = 151.9  # [s]

u_man = 0  # [W]
u_min = -3640.2 # [W]
u_max = 3640.2  # [W]

controller_params = (Kp, Ti, u_man, u_min, u_max)

# %% Preallocation of arrays for plotting etc:

t_array = np.zeros(N_sim)
Tin_sp_array = np.zeros(N_sim)
Tin_array = np.zeros(N_sim)
Qrad_array = np.zeros(N_sim)
Tout_array = np.zeros(N_sim)
u_array = np.zeros(N_sim)

# %% Simulation loop:
for k in range(0, N_sim):
t_k = k*dt   # Time
    # Setting input:
    if (t_k >= 0):
        Tin_sp_k = Tin_sp
    # PI controller:
    (u_i_k, u_k) = fun_pi(Tin_sp_k, Tin_k, u_i_km1,
                          controller_params, dt)
    # Process sim (Euler integration):
    Tin_kp1 = process_sim(Tin_k, u_k, process_params, dt)

    # Updating arrays for plotting:
    t_array[k] = t_k
    Tin_array[k] = Tin_k
    Tin_sp_array[k] = Tin_sp_k
    Qrad_array[k] = Qrad_k
    Tout_array[k] = Tout_k
    u_array[k] = u_k

    # Time shift:
    Tin_k = Tin_kp1
```

```
    u_i_km1 = u_i_k

# %% Plotting:
plt.close('all')
plt.figure(figsize=(7, 6))

plt.subplot(2, 1, 1)
plt.plot(t_array, Tin_sp_array,'m', label='Tin_sp')
plt.plot(t_array, Tin_array,'b', label='Tin')
plt.legend()
plt.xlabel('t [m]')
plt.ylabel('[oC]')
plt.grid()

plt.subplot(2, 1, 2)
plt.plot(t_array, u_array, 'r', label='Power u = P')
plt.plot(t_array, Qrad_array,'g', label='SolarIrrad')
plt.plot(t_array, Tout_array,'y', label='OutsideTemp')
plt.legend()
plt.grid()
plt.xlabel('t [m]')
plt.ylabel('[W]')

plt.subplots_adjust((hspace=0.4))
plt.ylim(0,3400)

plt.show()
```

Appendix 4. 5

```
# Varying On-OFF controller
import numpy as np
import matplotlib.pyplot as plt

# Greenhouse parameters
Crad = 0.1
Ag = 12.5
Qcoe = 0.21
Ac = 2.2
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
Kcoe = -7.88
Qven = 0.29

# Static input signals:
Tout_k = 10.0
Qrad_k = 108.2
```

```
Tin_init = 8  # [oC]
Tin_k = Tin_init  # [oC]

# Varying setpoint
Tin_sp = np.ones(60)*22.1 # [oC]
Tin_sp[10:30] = 21.0
Tin_sp[30:60] = 23.0

# Derived model params:
Qsun = Crad * Qrad_k *Ag
Qcov = Qcoe * (Tin_k - Tout_k) * Ac
Qsoil = Kcoe * (Ts - Tin_k) * Ag
Ccap = Rhoa * Ca * Vg
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k)

# Control input signal:
u_on = 3640.2 # [W]
u_off = -3640.2 # [W]
hysteresis = 0.5 # [oC]


# Simulate system response
Tin = [Tin_k]
u = [u_off]
for i in range(1, len(Tin_sp)):
    # Calculate model inputs
    Qsun_i = Crad * Qrad_k *Ag
    Qcov_i = Qcoe * (Tin[i-1] - Tout_k) * Ac
    Qsoil_i = Kcoe * (Ts - Tin[i-1]) * Ag
    Ccap_i = Rhoa * Ca * Vg
    Qvent_i = Qven * Rhoa * Ca * (Tout_k - Tin[i-1])

    # Calculate control action
    if Tin[i-1] < Tin_sp[i] - hysteresis:
        u_i = u_on
    elif Tin[i-1] > Tin_sp[i] + hysteresis:
        u_i = u_off
    else:
        u_i = u[-1]

    # Calculate system response
    Tin_i = Tin[i-1] + (Qsun_i - Qcov_i - Qsoil_i - Qvent_i + u_i) / Ccap_i

    # Save results
    Tin.append(Tin_i)
    u.append(u_i)
```

```
# To extract the data to a CSV file
import pandas as pd

# Initialize temperature and energy data lists
temperature_data = [Tin_k]
energy_data = [u_off]


# Simulate system response
Tin = [Tin_k]
u = [u_off]
for i in range(1, len(Tin_sp)):
    # Calculate model inputs
    Qsun_i = Crad * Qrad_k *Ag
    Qcov_i = Qcoe * (Tin[i-1] - Tout_k) * Ac
    Qsoil_i = Kcoe * (Ts - Tin[i-1]) * Ag
    Ccap_i = Rhoa * Ca * Vg
    Qvent_i = Qven * Rhoa * Ca * (Tout_k - Tin[i-1])

    # Calculate control action
    if Tin[i-1] < Tin_sp[i] - hysteresis:
        u_i = u_on
    elif Tin[i-1] > Tin_sp[i] + hysteresis:
        u_i = u_off
    else:
        u_i = u[-1]

    # Limit heating power if Tin exceeds 23 degrees Celsius
    if Tin[i-1] + (Qsun_i - Qcov_i - Qsoil_i - Qvent_i + u_i) / Ccap_i > 23:
        u_i = (23 - Tin[i-1]) * Ccap_i - Qsun_i + Qcov_i + Qsoil_i + Qvent_i

    # Calculate system response
    Tin_i = Tin[i-1] + (Qsun_i - Qcov_i - Qsoil_i - Qvent_i + u_i) / Ccap_i

    # Save results
    Tin.append(Tin_i)
    u.append(u_i)
# Save temperature and energy usage data
    temperature_data.append(Tin_i)
    energy_data.append(u_i)

# Create a Pandas DataFrame with the data
data = pd.DataFrame({'Temperature': temperature_data, 'Energy Usage':
energy_data})

# Export the data to a CSV file
data.to_csv('temperature_energy_dataOn-OFFNEW1.csv', index=False)
```

```
# Plot the results
plt.figure()
plt.plot(Tin_sp, 'k--', label='Setpoint')
plt.plot(Tin, 'b-', label='Tin')
plt.legend(loc='lower right')
plt.grid()
plt.xlabel('Time (m)')
plt.ylabel('Temperature [deg C]')
plt.twinx()
plt.plot(u, 'r-', label='Heating')
plt.ylabel('Heating Power [W]')
plt.legend(loc='lower left')
plt.show()
```

Appendix 4. 6

```
# Varying PI controller
import numpy as np
from gekko import GEKKO
import matplotlib.pyplot as plt

# Greenhouse parameters
Crad = 0.1
Ag = 12.5
Qcoe = 0.21
Ac = 2.2
Kcoe = -7.88
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
Qven = 0.29

# Static input signals:
Tout_k = 10.0
Qrad_k = 108.2

Tin_init = 8  # [oC]
Tin_k = Tin_init  # [oC]

# Varying setpoint
Tin_sp = np.ones(60)*22.1 # [oC]
Tin_sp[10:30] = 21.0
Tin_sp[30:60] = 23.0

# Control input signal:
u_min = -3640.2 # [W]
u_max = 3640.2  # [W]
```

```
# Derived model params:
Qsun = Crad * Qrad_k *Ag
Qcov = Qcoe * (Tin_k - Tout_k) * Ac
Qsoil = Kcoe * (Ts - Tin_k) * Ag
Ccap = Rhoa * Ca * Vg
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k)

# Initialize GEKKO model
m = GEKKO(remote=False)

# Time points
n_steps = len(Tin_sp)
m.time = np.linspace(0, n_steps-1, n_steps)

# Parameters
Qsun = m.Param(value=Qsun)
Qcov = m.Param(value=Qcov)
Qsoil = m.Param(value=Qsoil)
Ccap = m.Param(value=Ccap)
Qvent = m.Param(value=Qvent)

Kc = 132.41659297513655
tauI = 151.99171368404757

# Manipulated variable
u = m.MV(value=0, lb=u_min, ub=u_max)
u.STATUS = 1  # control input to be optimized

# Controlled variable
Tin = m.CV(value=Tin_k)
Tin.FSTATUS = 1  # receive measurement

# Setpoint
Tin_sp = m.Param(value=Tin_sp)

# PI controller with IMC tuning
Kp = Kc
tauI = tauI
tauD = 0.0
b = Kp/tauI
a = b*tauD

# PI equation
delta_t = m.time[1] - m.time[0]  # time step
u_p = m.Var(Tin_k)
u_i = m.Var(0.0)
m.Equation(u_i.dt() == b * (Tin_sp - Tin) - u_p / tauI)
m.Equation(u == u_p + u_i)
```

```
# Process model
m.Equation(Tin.dt() == (-Tin + Qsun - Qcov - Qsoil - Qvent + u_p + u_i) / Ccap)

# Setpoint tracking objective
m.Obj(10*(Tin - Tin_sp)**2)

# Set up solver options
m.options.IMODE = 6 # MPC
m.options.CV_TYPE = 1 #

# Solve the model
m.solve(disp=False)

# Get the optimized control inputs
u_opt = u.value

# Calculate temperature and energy use
Tin_values = Tin.value
energy_use = 0
for i in range(n_steps):
    energy_use += abs(u_opt[i]) * delta_t

# Save data to CSV file
with open('data.csvPINEW', mode='w') as data_file:
    data_writer = csv.writer(data_file, delimiter=',', quotechar="",
quoting=csv.QUOTE_MINIMAL)
    data_writer.writerow(['Time (m)', 'Temperature (deg C)', 'Energy Use (W)'])
    for i in range(n_steps):
        data_writer.writerow([i, Tin_values[i], abs(u_opt[i]) * delta_t])

# Plot the results
plt.figure()
plt.plot(Tin_sp, 'k--', label='Setpoint')
plt.plot(Tin, 'b-', label='Tin')
plt.legend(loc='lower right')
plt.grid()
plt.xlabel('Time (m)')
plt.ylabel('Temperature [deg C]')
plt.twinx()
plt.plot(u, 'r-', label='Heating')
plt.ylabel('Heating Power [W]')
plt.legend(loc='lower left')
plt.show()
```

Appendix 4. 7

```
# Varying PID controller
import numpy as np
```

```
from gekko import GEKKO
import matplotlib.pyplot as plt

# Greenhouse parameters
Crad = 0.1
Ag = 12.5
Qcoe = 0.21
Ac = 2.2
Kcoe = -7.88
Ts = 11.05
Rhoa = 1.29
Ca = 1000.0
Vg = 3.87
Qven = 0.29

# Static input signals:
Tout_k = 10.0
Qrad_k = 108.2

Tin_init = 8  # [oC]
Tin_k = Tin_init  # [oC]

# Varying setpoint
Tin_sp = np.ones(60)*22.1 # [oC]
Tin_sp[10:30] = 21.0
Tin_sp[30:60] = 23.0

# Control input signal:
u_min = -3640.2 # [W]
u_max = 3640.2  # [W]

# Derived model params:
Qsun = Crad * Qrad_k *Ag
Qcov = Qcoe * (Tin_k - Tout_k) * Ac
Qsoil = Kcoe * (Ts - Tin_k) * Ag
Ccap = Rhoa * Ca * Vg
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k)

# Initialize GEKKO model
m = GEKKO()

# Time points
n_steps = len(Tin_sp)
m.time = np.linspace(0,n_steps-1,n_steps)

# Parameters
Qsun = m.Param(value=Qsun)
Qcov = m.Param(value=Qcov)
```

```
Qsoil = m.Param(value=Qsoil)
Ccap = m.Param(value=Ccap)
Qvent = m.Param(value=Qvent)

Kc = 132.41659297513655
tauI = 15199171.368404757
tauD = 5.0

# Manipulated variable
u = m.MV(value=0, lb=u_min, ub=u_max)
u.STATUS = 1  # control input to be optimized

# Controlled variable
Tin = m.CV(value=Tin_k)
Tin.FSTATUS = 1  # receive measurement

# Setpoint
Tin_sp = m.Param(value=Tin_sp)

# PID equation
delta_t = m.time[1] - m.time[0]  # time step
u_p = m.Var(Tin_k)
u_i = m.Var(0.0)
u_d = m.Var(0.0)
m.Equation(u_d * delta_t == (tauD * (- u_p)) / Kc)
m.Equation(u_i.dt() == (u_i + delta_t / tauI * (Tin_sp - Tin)))
m.Equation(u == u_p + u_i + u_d)

# Process model
m.Equation(Tin.dt() == (-Tin + Qsun - Qcov - Qsoil - Qvent + u_p + u_i + u_d) /
Ccap)

# Initial condition for derivative term
u_p = m.Var(Tin_k)

# Setpoint tracking objective
m.Obj(10*(Tin - Tin_sp)**2)

# Set up solver options
m.options.IMODE = 6 # MPC
m.options.CV_TYPE = 1 #

# Solve the model
m.solve(disp=False)

# Get the optimized control inputs
u_opt = u.value
```

```
# Plot the results
plt.figure()
plt.plot(Tin_sp, 'k--', label='Setpoint')
plt.plot(Tin, 'b-', label='Tin')
plt.legend(loc='lower right')
plt.xlabel('Time (m)')
plt.grid()
plt.ylabel('Temperature [deg C]')
plt.twinx()
plt.plot(u, 'r-', label='Heating')
plt.ylabel('Heating Power [W]')
plt.legend(loc='lower left')
plt.show()

# Calculate energy use
energy_use = np.abs(u_opt) * delta_t

# Save time, temperature, and energy use data to a CSV file
with open('data.csvPID1NEW', mode='w') as file:
    writer = csv.writer(file)
    writer.writerow(['Time', 'Temperature', 'Energy Use'])
    for i in range(len(m.time)):
        writer.writerow([m.time[i], Tin[i], energy_use[i]])
```

Appendix 4. 8

```
# Varying MPC controller
from gekko import GEKKO
import matplotlib.pyplot as plt

# Greenhouse parameters
Crad = 0.1                  # Radiation transmission coefficient [-]
Ag = 12.5                   # Greenhouse surface area [m^2]
Qcoe = 0.21                 # Convective heat transfer coefficient [W/m^2 K]
Ac = 2.2                    # Cover surface area [m^2]
Kcoe = 96.3                 # Soil thermal conductivity [W/m K]
Ts = 11.05                  # Soil temperature [oC]
Rhoa = 1.29                 # Density of air [kg/m^3]
Ca = 1000.0                 # Specific heat of air [J/kg K]
Vg = 3.87                   # Greenhouse volume [m^3]
Kcoe = -7.88                # Soil heat loss [W/K m^2]
Qven = 0.29                 # Ventilation rate [m^3/s]

# Static input signals:
Tout_k = 10.0               # Outside air temperature [oC]
Qrad_k = 108.2              # Radiation heat gain [W/m^2]

# Varying setpoint
Tin_sp = np.ones(60)*22.1       # Indoor air temperature setpoint [oC]
```

```
Tin_sp[10:30] = 21.0
Tin_sp[30:60] = 23.0

# Control input signal:
u_min = -3640.2                    # Minimum heating power [W]
u_max = 3640.2                     # Maximum heating power [W]

# Derived model params:
Qsun = Crad * Qrad_k *Ag             # Solar radiation heat gain [W]
Qcov = Qcoe * (Tin_k - Tout_k) * Ac    # Convective heat loss through cover [W]
Qsoil = Kcoe * (Ts - Tin_k) * Ag       # Soil heat loss [W]
Ccap = Rhoa * Ca * Vg                # Air heat capacity [J/K]
Qvent = Qven * Rhoa * Ca * (Tout_k - Tin_k) # Heat loss through ventilation [W]

# Initialize GEKKO model
m = GEKKO()

# Time points
n_steps = len(Tin_sp)
m.time = np.linspace(0,n_steps-1,n_steps)
Tin_k = 8

# Parameters
Qsun = m.Param(value=Qsun)
Qcov = m.Param(value=Qcov)
Qsoil = m.Param(value=Qsoil)
Ccap = m.Param(value=Ccap)
Qvent = m.Param(value=Qvent)

# Manipulated variable (Control input)
u = m.MV(value=10, lb=u_min, ub=u_max)
u.STATUS = 1  # control input to be optimized

# Controlled variable
Tin = m.CV(value=Tin_k)
Tin.FSTATUS = 1  # receive measurement

# Setpoint
Tin_sp = m.Param(value=Tin_sp)

# Process model
m.Equation(Ccap * Tin.dt() == Qsun - Qcov - Qsoil - Qvent + u)
m.Obj((Tin - Tin_sp)**2) # setpoint weight#m.Obj

# Set up solver options
m.options.IMODE = 6  # MPC (tuning parameter)
m.options.CV_TYPE = 2  # Objective type (tuning parameter)
m.options.SOLVER = 2  # BPOPT (tuning parameter)
```

```
# Solve the optimization problem
m.solve(False)

# Get the optimized control inputs
u_opt = u.value

# Get the optimized control inputs
u_opt = u.value

u.DCOST = 0.001 # smooth out changes in control action

# Store temperature and energy usage data in CSV file
with open('data.csvMPCNEW', mode='w', newline='') as file:
    writer = csv.writer(file)
    writer.writerow(['Time (m)', 'Temperature [deg C]', 'Heating Power [W]'])
    for i in range(n_steps):
        writer.writerow([i, Tin_sp[i], Tin[i], u_opt[i]])

# Plot the results
plt.figure()
plt.plot(Tin_sp, 'k--', label='Setpoint')
plt.plot(Tin, 'b-', label='Tin')
plt.grid()
plt.legend(loc='lower left')
plt.xlabel('Time (m)')
plt.ylabel('Temperature [deg C]')
plt.twinx()
plt.plot(u, 'r-', label='Heating')
plt.ylabel('Heating Power [W]')
plt.legend(loc='lower right')
plt.show()
```

Appendix 5. 1

IoT control system – Real time decision with PID

The things data program algorithm structure of PID

```
/*
  Sketch generated by the Arduino IoT Cloud Thing "FidelityThermocouple3"
  https://create.arduino.cc/cloud/things/c8c0560c-b3e7-4bf5-b7d4-
b4e705aa633c
  Arduino IoT Cloud Variables description
  The following variables are automatically generated and updated when changes
are    made to the Thing

float thermo3;
  int relay3;
```

```
  Variables which are marked as READ/WRITE in the Cloud Thing will also have functions
  which are called when their values are changed from the Dashboard.
  These functions are generated with the Thing and added at the end of this sketch.
*/
#include "thingProperties.h"
// sensor&relay parameters/library
#include "max6675.h"
int soPin = 12;               // SO=Serial Out
int csPin = 10;               // CS = chip select CS pin
int sckPin = 13;              // SCK = Serial Clock pin
int OutputPin = 3;            // relay pin

// PID Parameters
double Kp = 2.04;             //proportional gain
double Ki = 3.41;             //integral gain
double Kd = 0.306;            //derivative gain
int T = 50;                   //sample time in milliseconds (ms)

unsigned long last_time;
double total_error, last_error;
float temperatureC;
double setpoint;
int sensorVoltage;
double voltage;

MAX6675 beatrice(sckPin, csPin, soPin);// create instance object of MAX6675
void setup() {
  // Initialize serial and wait for port to open:
  Serial.begin(9600);
  delay(1500);

  pinMode(OutputPin, OUTPUT);
  Serial.println("Beatrice MAX6675");

  // Defined in thingProperties.h
  initProperties();

  // Connect to Arduino IoT Cloud
  ArduinoCloud.begin(ArduinoIoTPreferredConnection);
   /*
     The following function allows you to obtain more information
     related to the state of network and IoT Cloud connection and errors
     the higher number the more granular information you'll get.
     The default is 0 (only errors).
     Maximum is 4
 */
```

```
  setDebugMessageLevel(2);
  ArduinoCloud.printDebugInfo();
}

void loop() {
  ArduinoCloud.update();

  // Your code here
  thermo3 = (beatrice.readCelsius());

  Serial.print("C = ");
  Serial.print(beatrice.readCelsius());
  Serial.print(" F = ");
  Serial.println(beatrice.readFahrenheit());

  delay(1000);
  PID_Control();
}
  void PID_Control() {
  thermo3 = (beatrice.readCelsius());
  setpoint = 23;

  analogWrite(OutputPin, relay3);                    //sends PWM signal, duty cycle
set between 0 and 255;

  unsigned long current_time = millis();
  int delta_time = current_time - last_time;

  if (delta_time >= T) {

double error = setpoint - thermo3;
total_error += error; //accumalates the error
    if (total_error >= 255) total_error = 255;
    else if (total_error <= 0) total_error = 0;

    double delta_error = error - last_error; //difference of error(current error - last
error)
    relay3 = Kp * error + (Ki * T) * total_error + (Kd / T) * delta_error; //PID control
compute
    if (relay3 >= 255) relay3 = 255;
    else if (relay3 <= 0) relay3 = 0;
    last_error = error;  // after the computation, we store the error as last error for
the   next iteration
    last_time = current_time;  // same forthe time
  }

 }
/*
```

```
  Since Relay3 is READ_WRITE variable, onRelay3Change() is
  executed every time a new value is received from IoT Cloud.
*/
void onRelay3Change()  {
  // Add your code here to act upon Relay3 change

 }
```

Greenhouse 25th May Data

Time(s)	Input (u)	Output (y)
6.192	255	17.5
32.621	0	18.25
47.091	0	18
27.467	86	18.25
28.517	0	17.75
42.197	86	18
43.535	42	18.25
44.277	0	17.75
49.537	43	18.25
50.594	0	17.75
56.897	43	18.25
57.947	0	18.5
0.257	430	18.25
1.117	0	18
2.157	43	18.25
3.337	0	17.5
11.627	43	18.25
12.677	85	18
15.837	42	18.25
19.007	85	18
20.047	42	18.5
21.097	0	18.25
23.207	43	18
24.267	42	18.25
27.427	85	18
28.507	42	18.25
29.537	85	18.5
30.597	128	18.25
31.647	127	18.5
35.887	84	18.25
36.927	85	18.75
37.987	128	18.25
39.177	127	18.5

43.257	171	18.25
44.317	255	18.5
26.667	212	18.75
27.717	83	18.5
28.767	0	18.75
14.267	43	18.5
15.317	0	18.75
14.037	43	18.5
15.151	0	18.75
17.187	43	19
18.267	0	18.75
32.917	43	19.25
33.967	0	18.25
44.638	43	19.25
45.518	85	19
47.637	42	18.75
48.677	0	19.5
51.837	43	19.25
52.887	0	19.5
1.297	43	19.25
2.357	42	19.5
3.387	0	19.25
8.667	43	19.5
9.697	0	19.75
13.897	43	19.5
14.957	42	19.75
16.007	128	19.5
17.068	127	20.25
21.277	84	19.5
22.337	0	20
35.967	86	19.5
37.047	128	20.25
38.087	84	20.5
39.127	0	20.75
51.737	43	20
52.778	42	20.5
53.827	85	20.75
54.929	128	20.5
55.947	127	20.75
56.998	171	20.5
58.127	127	21
0.167	84	20.75
1.217	85	21.5

3.337	171	21
4.388	170	21.5
7.547	213	20.75
8.607	169	21.75
9.677	170	21.25
10.717	127	21.75
11.787	84	21
12.907	85	21.5
16.187	128	22
17.047	127	22.25
19.157	171	21.75
20.207	213	22.25
21.274	255	21.75
22.317	212	22.5
23.387	169	22.25
24.427	213	22.5
26.547	255	22
28.657	212	22.75
29.717	213	22.5
32.877	255	22.25
56.744	212	22.5
57.345	169	22.75
58.167	0	22.5
32.547	43	23.25
33.587	0	22.75
48.281	43	23.5
49.447	0	23.25
37.387	43	23.75
38.437	0	24
31.957	43	23.75
32.997	85	24
36.167	42	24.25
37.207	0	24
40.357	43	24.25
41.407	0	23.5
43.517	43	24
44.577	0	24.25
49.837	43	24.5
50.887	0	24.25
9.768	43	24.75
10.827	85	24.5
11.877	128	24.75
13.017	127	25

19.267	84	25.25
20.317	42	24.75
21.387	85	25
23.487	42	25.25
25.715	0	25
30.855	43	25.5
32.057	42	25.75
32.957	0	26
36.117	129	25.75
37.397	171	26.25
38.237	84	26.5
39.277	85	26.25
41.387	0	26.5
45.577	43	26.75
46.637	42	26.5
47.717	85	26.75
48.757	42	26.5
49.807	85	27
50.877	42	27.25
51.935	0	27.5
56.117	43	27.25
57.187	42	27.5
59.277	0	27.75
2.427	43	28
3.487	42	28.25
5.597	85	27.75
6.647	42	28.5
8.757	0	28
15.077	86	28.25
16.297	85	28.5
18.237	42	28.75
21.397	85	28
22.467	42	28.75
23.527	85	28.5
26.677	171	28.75
27.754	170	29
30.907	255	29.25
36.187	212	28.75
37.236	169	29.25
38.288	213	29.5
41.467	126	29.75
42.507	127	29.5
44.866	214	29.75

45.687	169	30
46.747	213	30.5
50.137	169	30.25
50.957	170	30.5
52.137	213	30
54.126	169	30.25
55.227	127	30.5
0.466	171	30.75
1.526	170	30
3.616	127	31
7.855	171	30.75
8.887	170	30.5
10.996	84	31.25
12.066	85	31
14.176	42	31.25
16.276	85	31
18.397	42	30.75
19.456	0	31
27.98	43	30.75
28.906	0	31.25
30.026	43	31
31.066	0	30.75
54.187	43	31.25
55.136	0	31.75
58.192	43	31.5
59.136	0	31.25
42.586	43	31
43.636	0	31.5
50.986	43	30.75
52.06	0	31.75
57.276	86	32
58.346	171	31.5
59.406	84	32
0.456	85	31.75
1.506	0	32
5.716	43	32.25
6.764	0	31.5
53.825	43	32
54.746	0	31.75
33.545	43	32
34.596	42	32.25
35.646	85	32
36.716	42	32.5

37.766	0	32.25
43.003	43	32.5
44.075	0	32.25
50.335	129	32.5
51.395	84	33
52.456	42	32.5
54.576	0	33.25
56.906	43	33
57.715	42	32.75
58.766	0	33.25
1.926	43	33.5
2.976	85	33
4.036	128	33.75
5.086	171	33
6.156	170	33.5
7.365	127	32.75
10.626	171	33.75
11.445	170	33.5
12.476	213	33.75
13.546	255	33.25
10.655	125	33.5
11.715	0	33.25
2022-05-25T11:40:11.715Z	0	33.25