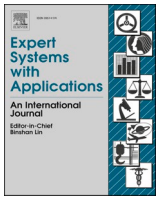
Expert Systems With Applications 252 (2024) 124126

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



Deep reinforcement learning for PID parameter tuning in greenhouse HVAC system energy Optimization: A TRNSYS-Python cosimulation approach

Misbaudeen Aderemi Adesanya [a], Hammed Obasekore [b], Anis Rabiu [a], Wook-Ho Na [c], Qazeem Opeyemi Ogunlowo [a,d], Timothy Denen Akpenpuun [c,e], Min-Hwi Kim [f], Hyeon-Tae Kim [g], Bo-Yeong Kang [b], Hyun-Woo Lee [a,c,*]

[a] Department of Agricultural Civil Engineering, College of Agricultural and Life Sciences, Kyungpook National University, Daegu 41566, Republic of Korea
[b] Department of Robot and Smart System Engineering, Kyungpook National University, Daegu 41566, Republic of Korea
[c] Smart Agriculture Innovation Centre, Kyungpook National University, Daegu 41566, Republic of Korea
[d] Department of Agricultural and Bioenvironmental Engineering, Federal College of Agriculture Ibadan, PMB 5029, Ibadan, Nigeria
[e] Department of Agricultural and Biosystems Engineering, University of Ilorin, PMB 1515, Ilorin 240103, Nigeria
[f] Renewable Energy System Laboratory, Korea Institute of Energy Research, 152 Gajeong-ro, Yuseong-gu Daejeon 34129, Republic of Korea
[g] Department of Bio-Industrial Machinery Engineering, Gyeongsang National University, Jinju 52828, Republic of Korea

ARTICLE INFO

Keywords:
HVAC control
TRNSYS
Python
Optimization
Deep reinforcement learning
Cosimulation

ABSTRACT

The control of indoor temperature in greenhouses is crucial as it directly impacts the crop's thermal comfort and the performance of heating, ventilation, and air-conditioning (HVAC) systems. Conventional feedback controllers, like on/off, can sometimes make HVAC system work at full capacity when only half that capacity is needed. In contrast, the proportional-integral-derivative (PID) controller, provides precise control based on its P, I, and D parameters. However, it lacks a formal design procedure for optimizing a specified objective function. Previous studies have utilized conventional PID tuning approaches to track room setpoint temperature for residential buildings, data centers, and office buildings, with limited research in greenhouse applications. To address this gap, this study proposes a flexible PID controller that employs a deep reinforcement learning (DRL) algorithm to optimize its parameters, by tracking the setpoints and energy consumption of a greenhouse planted with tomatoes. This approach is different from the typical method of using the trained RL agent directly in HVAC controls. Through a self-made TRNSYS-Python cosimulation framework, the DRL agent interacts directly and in real time with the greenhouse and its plants. Consequently, optimized PID parameters were established and tested in the simulated environment. The resulting performance, in terms of both energy consumption and its ability to maintain the crop's comfort temperature, was compared with the simulated on/off and manually tuned PID controllers. Compared to the on/off baseline control, the proposed PID optimized parameters reduce energy use by 8.81% to 12.99% and the manually tuned PID parameters with the Ziegler-Nichols tuning method reduce energy use by 7.17 %. Additionally, the proposed method had a deviation of 2.07% to 3.13%, while the manually tuned PID controller and the on/off controller had deviations of 7.27% and 3.27%, respectively, from the minimum comfortable temperature. This study serves as a framework for improving the energy efficiency of greenhouse HVAC system operations.

Abbreviations: BIA, Building-integrated agriculture; DDPG, Deep Deterministic Policy Gradient; DRL, Deep Reinforcement Learning; DNN, Deep Neural Network; DQN, Deep Q-Network; e, Error; Engineering Equation Solver, EES; ESP-r, Environmental Systems Performance-Research; FCU, Fan-coil units; HVAC, Heating, ventilation, and air-conditioning; HWP, Hot-water pipe; K_D, Derivative gain; K_I, Integral gain; K_P, Proportional gain; l, Loss; MDP, Markov Decision Process; ML, Machine learning; MPC, Model predictive control; NSE, Nash–Sutcliffe coefficient; PER, Prioritized Experience Replay; PID, Proportional integral differential; PSF, Purme Social Farm; RH, Relative humidity; RL, Reinforcement learning; RMSE, Root mean square error; RMPC, Robust model predictive control; SR, Solar radiation; TD, Temporal difference, TD3, Twin Delay Deep Deterministic Policy Gradient; TRNSYS, TraNsient System Simulation.

* Corresponding author.
E-mail addresses: misbauadesanya@knu.ac.kr (M.A. Adesanya), obasekore.hammed@knu.ac.kr (H. Obasekore), rabiuanis@knu.ac.kr (A. Rabiu), wooks121@knu.ac.kr (W.-H. Na), ogunlowoqazeem@knu.ac.kr (Q.O. Ogunlowo), akpenpuun.td@unilorin.edu.ng (T.D. Akpenpuun), mhkim001@kier.re.kr (M.-H. Kim), bioani@gnu.ac.kr (H.-T. Kim), kby09@knu.ac.kr (B.-Y. Kang), whlee@knu.ac.kr (H.-W. Lee).

https://doi.org/10.1016/j.eswa.2024.124126
Received 30 November 2023; Received in revised form 13 April 2024; Accepted 26 April 2024
Available online 28 April 2024

1. Introduction

The exponential growth of the human population and the seasonal unpredictability of climate conditions have underscored the need for controlled-environment agriculture to ensure year-round food production and maintain optimal growing conditions for crops. This has led to the rise of greenhouses, vertical farms, and building-integrated agriculture (BIA). Greenhouses, in particular, harness more of the sun's energy for photosynthesis and heating compared to vertical farms and BIA (Gao et al., 2024). However, when outside temperatures exceed 30 °C or drop below 10 °C, greenhouses require heating and/or cooling using heating, ventilation, and air-conditioning (HVAC) systems to support crop growth (Baudoin et al., 2013).

Aside from the conventional on/off control, the proportional-integral-derivative (PID) method is widely used in controlling HVAC systems. In most greenhouse HVAC systems, a PID controller is already in place, with its role being to minimize the error between the setpoint and the measured value. The actual control effort involves multiplying the error by a proportional gain (K_P), integrating the error by an integral gain (K_I), and taking the derivative of the error by a derivative gain (K_D), as depicted in Fig. 1. Achieving optimal tracking requires setting the K_P, K_I, and K_D parameters to match the system's dynamics through tuning. While conventional tuning techniques have been applied in controlling residential buildings (Ghaddar et al., 2021), offices (Copot et al., 2018; Soyguder et al., 2009), and data centers (Durand-Estebe et al., 2013), their application in greenhouses remains limited. Moreover, the optimal tuning of PID gains to minimize setpoint error and energy consumption through reinforcement learning (RL) deviates from the usual practice of using a trained RL agent in controlling HVAC system.

Machine learning (ML), a field that has found practical applications in various domains, is generally categorized into supervised, unsupervised and RL. Each type relies on data, but the nature of the data differs. Supervised learning, for instance, is about learning from a collection of input–output data pairs, resulting in a mapping function that predicts outputs for new input data. Unsupervised learning, on the other hand, learns the underlying pattern in just input data. In between, a subtle type called semi-supervised learning has been recognized; it is a situation where the learning data contains little output. As for RL, its focuses on the idea that direct stepwise interaction with the data source (usually a Markov Decision Process (MDP)-based environment for the dynamics of the system) to subsequently get rewards or punishments because of actions taken on the environment. Using these observed rewards, the agent is expected to learn an optimal (or nearly optimal) policy for the environment to produce actions that can obtain maximum benefits. In complex domains such as agriculture, RL stands out as the most viable method to train a policy to perform at a competitive level, even matching human heuristics.

Formally, RL is inherently formulated to directly interact step-wisely with an MDP-based environment. At each step, the agent takes actions (a_t) based on the observed state (s_t) and subsequently gets a reward (r_t), thus transitioning the environment to the next state (s_{t+1}) (Russel and Norvig, 2010). Using these observed rewards, the agent is expected to learn an optimal (or nearly optimal) policy for the environment. Fig. 2 shows an overview of the RL paradigm. This policy is a rule used by an agent to determine the best action in each state. It can be represented as a look-up table or a universal function approximator, such as a Q-Table or Q-Network, respectively. In either case, the Q-learning algorithm is responsible for the iterative update of the policy using the Bellman Eq. (1) which is a function of the state and action.

$$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha(r_t(s) + \gamma \cdot \max Q(s_{t+1}, a)) \quad (1)$$

where α and γ are the learning rate and discounting factor that takes a value in the range of 0 and 1, the learning rate is a hyperparameter that is responsible for the rate at which the algorithm converges. However, this parameter needs to be carefully selected to avoid oscillation when it's too large or slow convergence when too small. The discounting factor parameter sets priority for short- or long-term reward. For instance, setting the discounting factor to 0.99 means that the agent will prioritize actions that lead to higher rewards in future.

Recent RL breakthrough introduces the possibility of utilising deep neural network (DNN) as the Q – function that estimates the Q – values end-to-end. Thus, launching the innovative area of Deep Reinforcement Learning (DRL) (Mnih et al., 2013). DRL has proven successful in handling a wide range of more complex and challenging tasks with high-level inputs, like RGB image, over the Q-Table. A few of these tasks

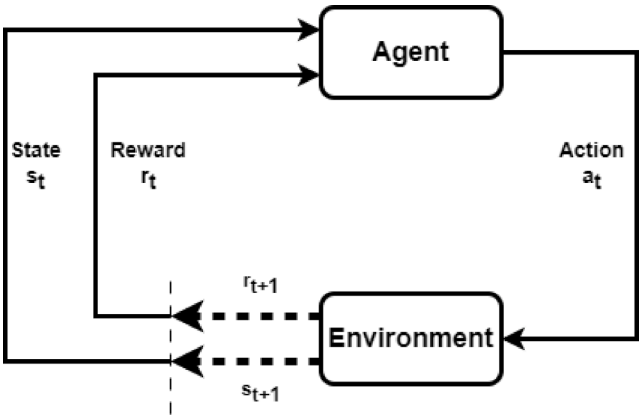


Fig. 2. Schematic process of RL operation.

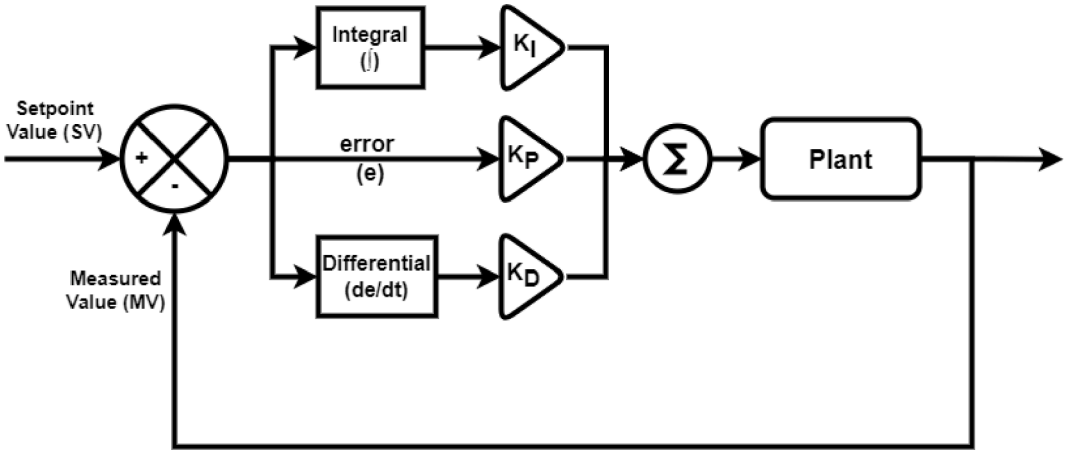


Fig. 1. The structure of a PID controller for greenhouse application.

include gaming tasks (Mnih et al., 2013), autonomous driving (Kiran et al., 2022), and robotic control tasks (Jeon et al., 2024). This success underscores the potential of DRL in tackling complex real-world problems like in the case of Greenhouse HVAC optimisation.

While Deep Q-Network (DQN) happens to be the maiden breakthrough, several derivatives have emerged afterwards, such as Dueling DQN (Wang et al., 2016), Double DQN (Van Hasselt et al., 2016), Rainbow (Hessel et al., 2018), Prioritized Experience Replay (PER) (Schaul et al., 2015). But generally, the role of the DQN algorithm is to minimize a sequence of expected loss, $L_i(\theta_i)$, for a Q-Network with weight θ as shown in Eq. (2).

$$L_i(\theta_i) = \mathrm{E}_{s,a,r,s_{t+1}\ \rho(.)}\left[(y_i - Q(s,a;\theta_i))^2\right] \tag{2}$$

where $y_i = r + \gamma \cdot \max a \cdot Q(s,a;\theta_{i-1})$, that is the TD (temporal difference) target, and y − Q is called the TD error. $\rho(.)$ represents the behavior distribution, the distribution over transitions {s, a, r, s_{t+1}} collected from the environment. Furthermore, it also uses two major components that significantly stabilize learning, namely, experienced replay, policy and target network. The experience replay is a buffer memory (D) that stores the agent's decorrelated transitions {s, a, r, s_{t+1}} at every timesteps. During training, instead of using just the latest transition (as in the case of Q – learning) to compute the loss and gradient of the network, randomly sampled transitions from the replay buffer are used. Additionally, DQN distinguishes between policy and target network. Policy network is the main network that interacts with the environment to select actions based on the current state, while the target network is a snapshot of the network parameters from a few iterations ago instead of the last iteration. Usually, it has the same architecture as the policy network but with frozen parameters for a certain number of iterations (Mnih et al., 2013).

Intriguing, RL agent requires direct interaction with a system to build its state–action, thus necessitating the need to access real-time data measurements from sensors and actuation control. Many applications of RL in controlling HVAC systems used historical weather data after learning from sufficient training to adjust to the prevailing climatic conditions (Ajagekar et al., 2023; Qin et al., 2023). However, before the RL algorithm converges, predicted actions and frequency of sensor data reading are not always safe for real systems. Therefore, a state-space model that accurately describes the dynamics of the physical system is preferred.

Numerical modeling software like IDA indoor climate energy, Engineering Equation Solver (EES), EnergyPlus, Environmental Systems Performance-Research (ESP-r), MATLAB, Modelica, and TraNsient System Simulation (TRNSYS) software can offer alternatives for ML models, specifically RL agents to learn about physical systems and create its state-space model. TRNSYS, unlike others, is a dynamic simulation tool that enables the cosimulation of complex interactions within several simulation tools and integration of other surrogate models. The TRNSYS simulation tool allows for modeling systems using standard library components or user-defined nonstandard components called "Types" that can be written in C++, FORTRAN, or Python (Ogunlowo et al., 2022). TRNSYS facilitates the development of a state-space model for greenhouses, employing approximation methods to represent heat flow in various areas based on envelope properties and external weather conditions (Rabiu et al., 2023). The major setback to using the TRNSYS model as an RL environment is its inability to allow the RL algorithm to perform stepwise interaction within the simulation contrary to its uninterruptible start-to-finish configuration, abating HVAC system to use optimized controls from RL agent.

Therefore, in this paper, we implement a framework that leverages on an RL agent to optimize the energy consumption of a previously validated TRNSYS greenhouse model by Adesanya et al. (2022). In using the model as the RL environment, TRNSYS Type 3157 (Bernier et al., 2022), which permits an event-driven Python call from TRNSYS, was utilized to write the observation, and read the action (control and energy consumption). Another external Python shell wrapped the model as an OpenAI gym (Brockman et al., 2016) environment, to read observations, compute rewards, and write actions from the RL agent to control the TRNSYS simulation timestep. Considering that the greenhouse air temperature is a continuous variable whose value is dependent on the operation of the HVAC system, the DQN algorithm was used in this study to solve the fixed control problem of TRNbuild heating type manager. The main contributions of this study include:

- Presents the design and implementation of a novel PID controller in a greenhouse equipped with an HVAC system.
- Proposes a method to vary the setpoint temperature of the TRNbuild heating type manager based on optimized PID parameters.
- Introduces an RL algorithm as a proxy to optimize PID parameters based on action intervention from the greenhouse environment.
- Develops a method to control the TRNSYS simulation time step based on updated control from the RL agent compared to uninterrupted running with a fixed control.

The paper's structure is outlined as follows: Section 2 provides a comprehensive review of relevant literature. In Section 3, the methodology of the proposed architecture is delineated, beginning with the study's assumptions, followed by the description of the experimental greenhouse, the cosimulation environment framework, and the statistical index for model evaluation. The cosimulation environment framework details the modelling procedure in TRNSYS, the self-tuning procedure of the PID parameters, the adoption of the DQN algorithm, the formulation of TRNSYS as an RL environment, the selection of observations, the setting of actions, and the formulation of the reward function for the RL environment. Section 4 presents the simulation and experimental results, including climate conditions during the investigation period, analysis of greenhouse energy efficiency, and comparative analysis of greenhouse temperature with different control methods. Section 5 discusses the study's insights, limitations, and future research focus. Finally, Section 6 concludes the study.

2. Related work

Precisely controlling HVAC systems in buildings is crucial as it directly impacts indoor air quality and energy consumption. Several control systems strategies have been developed, for instance, the classical control like bang-bang and PID control, modern control like fuzzy logic and rule-based control (RBC) (Brandi et al., 2020), optimal control like model predictive control (MPC) (Drgoňa et al., 2020), and linear quadratic regulator (LQR) control (Asad Rizvi et al., 2023). ML techniques, such as RL, have been claiming more futuristic performance in recent years. However, the adoption in real-world systems is limited due to its computational demand. Hence, most real-world systems rely heavily on classical techniques because of their low computational demand, precisely, on/off and PID control.

Research has shown the possibility of getting optimized parameters of the PID controller with artificial neural networks–based self-tuning, as proposed by Zhang et al. (2016). Despite the simplicity and light computational demand of PID, this tuning approach is not designed to handle multi-objective and highly nonlinear time-varying systems like a greenhouse with growing crops. Therefore, more robust techniques like the MPC (Afram et al., 2017) and various types of ML techniques have been employed. For instance, Hu and You (2022) delved into the dynamics of greenhouse and crop growth, considering factors like greenhouse geometry, weather dynamics, and control signals, to formulate a state-space model for MPC. Subsequently, they devise a data-driven Robust Model Predictive Control (RMPC) framework to regulate greenhouse temperature, without focusing on energy consumption.

In a quest to strike a balance between the already provisioned classical control and the excellent performance of ML in real-world HVAC systems, DRL is utilized in optimal control of HVAC systems and

scholastic works exist for the optimal control of HVAC systems with DRL agents. Brandi et al. (2020) compared a DRL algorithm with RBC and climatic-based logics control to attain a setpoint water supply temperature to a heating system. The DRL agent reduced the total energy consumption between 5 % and 12 % with an enhanced indoor temperature control with static and dynamic deployment. Similarly, an RL deep deterministic policy gradient (DDPG) algorithm has been reported to efficiently minimize energy consumption in residential HVAC systems while meeting indoor thermal comfort requirements (Gao et al., 2020; Du, Li, et al., 2021; Du, Zandi, et al., 2021). Likewise, an asynchronous advantage actor–critic algorithm has been employed for a data center (Biemann et al., 2021). Addressing energy consumption and microclimate conditions in the greenhouse, a Q-learning approach effectively handled continuous action space, resulting in a 61 % reduction in energy consumption compared to DDPG (Ajagekar & You, 2022). Seemingly, Ajagekar et al., (2023) integrated DRL techniques for greenhouse heating system control, achieving a 57 % reduction in energy consumption compared to MPC, RMPC, and DDPG.

Utilizing the open-source BOPTEST building optimization tool as a virtual environment, Gao and Wang (2023) examined the performance of model-based RL and model-free RL in controlling an HVAC system. The model-based RL version of the two model-free RLs, DQN and actor-critic, accelerated the learning of RL agents and achieved comparable control performance efficiency in a shorter training time. In the same vein, Shin et al. (2024) proposed a model-free RL control of HVAC systems based on weather forecasting data to reduce predictive error. They utilized a gated recurrent unit model to predict future outside weather data and employed experience replay technique combined with DQN for optimal HVAC system operation in EnergyPlus simulation, resulting in a 58.79 % energy savings compared to RBC. In a study by Qiu et al. on an energy system in the Nantong Central Innovation District in China, an improved RL based on Twin Delayed Deep Deterministic Policy Gradient (TD3) reduced system operating costs by 2.76 % compared to DDPG.

Azuatalam et al. (2020) compared EnergyPlus baseline with RL for HVAC energy reduction in a commercial building, with RL achieving a 12 % reduction in weekly energy usage. Also, Du et al. (2021) compared the performance of RBC and a model-free DRL based on the DDPG algorithm, with DDPG-based HVAC control reducing comfort violations by 98 %. Similarly, Solinas et al. (2024) investigated three RL approaches to HVAC optimization, finding that the direct online approach provided a reliable solution to HVAC system optimization while maintaining thermal comfort. However, the white-box model approach resulted in a 65 % energy consumption reduction and a 25 % improvement in thermal comfort compared to the baseline. Seemingly, Asad Rizvi et al. (2023) applied the Q-learning algorithm to LQR of a multi-zone HVAC system but observed algorithm bias due to disturbances from unknown heat gains, light sources, occupancy density, and outside weather variations. To address this, a lumped bias term was developed to enhance HVAC system control.

While extensive research has been conducted on various TRNSYS HVAC system components, the utilization of RL for precise control remains relatively unexplored. For instance, Żabnieńska-Góra et al. (2021) investigated internal and external heating systems for a low-energy single-family building in London. Their study employed a TRNbuild heating type manager and a PV/T system connected to an idealized mechanical ventilation system within TRNbuild. While the internal heating system accurately depicted the building's annual energy consumption, the PV/T system encountered difficulties in maintaining comfortable temperatures during winter. In a separate exploratory study, Prieto et al. (2023) modelled the thermal demand of a greenhouse in a Mediterranean climate in southern Spain using TRNbuild. Their energy supply system incorporated an integrated solar thermal system, biomass, and an air-cooled absorption chiller to regulate the greenhouse temperature for tomato cultivation.

3. Materials and methods

3.1. Study's assumptions

The study's first assumption involved utilizing the TRNbuild heat type manager to represent the HVAC system, including fan-coil units (FCU) and hot-water pipes (HWP), installed at the experimental site. Previously, Adesanya et al. (2024) presented the design of an optimal HVAC system based on estimated heat loads from TRNbuild and compared the experimental performance of the case study HVAC systems with several emission standards. However, this study advances the building model to operate on three heating scenarios using the heating type manager in TRNbuild. The real HVAC system to mimic the operation of the installed FCU and HWP could not be modelled in the TRNSYS simulation interface due to the absence of a TRNSYS-type FCU model based on manufacturers' data (Calise et al., 2012). Nevertheless, the TRNbuild heating type manager has proven accurate in predicting the energy consumption of greenhouse HVAC systems (Rasheed et al., 2022). Secondly, the use of file lock to synchronize simulation steps for observations and actions exchange assumed that only one instance of TRNSYS is interacting with the project environment during the experiment. Finally, depending on the workload on a computer, the operating system (OS) sets priority for program execution, potentially leading to premature termination of some processes. We considered such OS interruptions on TRNSYS as a terminal state and counted them as the end of the episode.

3.2. Description of the experimental greenhouse

The experimental site, Purme Social Farm (PSF), is located in Ohak-dong, Yeoju-si, Gyeonggi-do, South Korea as shown in Fig. 3 and according to Köppen's climate classification, the region possesses a monsoon-influenced hot-summer humid continental climate with an average annual temperature of 12.8 °C. To produce as much energy as the greenhouse consumes, PSF is integrated with solar thermal collectors, photovoltaic thermal collectors, heat pumps, and thermal energy storage systems. Also, it is equipped with HVAC terminal units that absorb thermal energy from heat pumps in the form of hot and cold water stored in short-term thermal energy storage and separated into two primary insulated headers: U-tube HWP and FCU. The HWP header provides the tube rails, growing tubes, side, and rear tubes as shown in Fig. 4. The tube rails are installed in 34 sections laterally on the ground floor to cover the greenhouse growing area. The greenhouse has an area of 3942 m^2: Farm A (2160 m^2) and Farm B (1782 m^2). Farms A and B are Venlo-roofed with dimensions of 32 m × 67.5 m × 7.25 m and 36 m ×

Fig. 3. Aerial view of the experimental greenhouse.

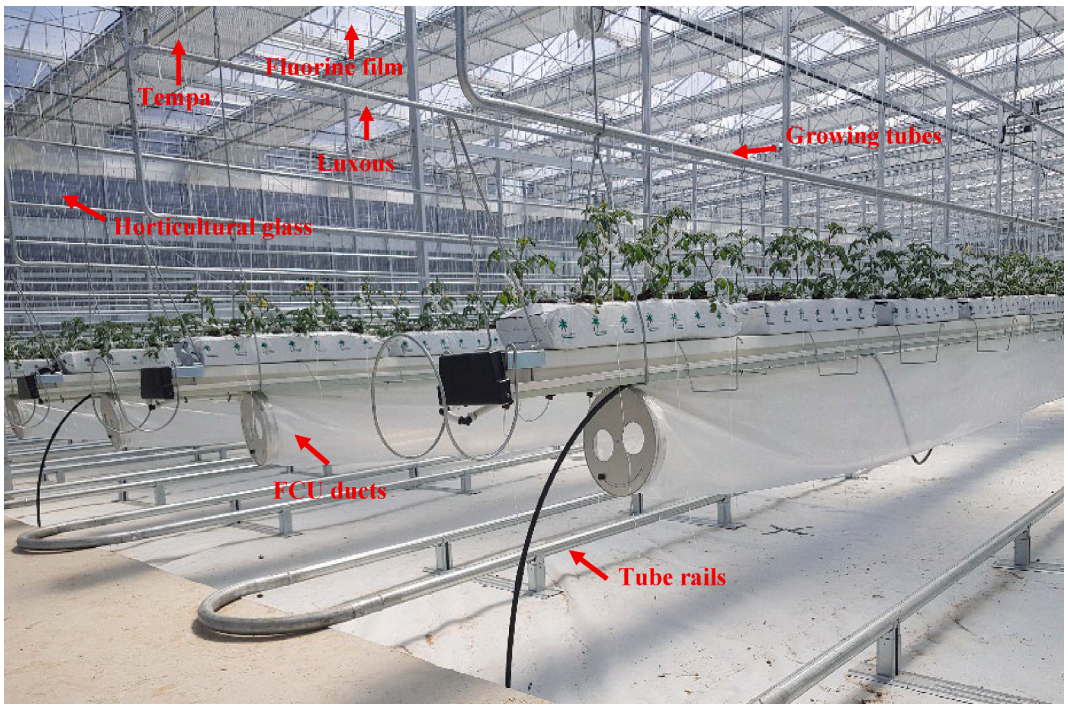


Fig. 4. Arrangement of the HVAC systems and positioning of the thermal screens.

49.5 m × 7.25 m, respectively.

To save energy, PSF was built with thermal screens and covering materials that are good at keeping heat. Horticultural glass and a fluorine film enclosed the sides and roof of the greenhouse, while retractable horizontal (Tempa and Luxous) and vertical (Obscura) thermal screens were installed inside the greenhouse. The white surface of the Obscura reflects light onto the crop to promote crop growth. Tempa is used in the summer to lessen daytime sun exposure. By diffusing and reflecting sunlight, it reduces the heat load of the greenhouse and keeps it cool (Rabiu et al., 2022). Luxous is a thermal screen that helps the greenhouse retain a steady temperature by holding heat at night. To measure the environmental conditions and energy consumption in the greenhouse, several sensors were installed in both farms. The properties of the installed sensors are listed in Table 1 and Eq. (3) was used to calculate the experimental greenhouse energy consumption.

$$Q = \dot{m} \times C_p \times \Delta t \tag{3}$$

where Q is the amount of energy consumed (kWh); $\dot{m}$ is the mass flow rate (L/min); C_p is the specific heat capacity of water (kJ/kg.C), and Δt is the changes in water temperature (C).

3.3. Cosimulation environment framework

3.3.1. Modelling in TRNSYS

The greenhouse modelling and simulation process involved several steps. Initially, the greenhouse was designed in TRNSYS 3D, an add-on of Google Sketchup, as multi-zones with 17 spans of 4 m each, a ridge height of 0.95 m, and a roof slope of 30°. Farms A and B are represented by two of the zones, with eight and nine spans, respectively, as shown in Fig. 5. The entire surfaces of the zones were modeled as a window to define the exact greenhouse material properties using the Lawrence Berkeley National Laboratory Windows 7.7 program. The 3D model was then imported into TRNbuild (a building interface of TRNSYS component named Type 56) as a.idf file.

Table 1
Sensor specifications.

Parameter	Unit	Sensor	Sensor Precision
Ambient temperature	°C	HOBOMX1102A	±0.5°C
Relative humidity	%	HOBOMX1102A	±0.5%
Solar radiation	Wm^{-2}	CMP3 pyranometer	±20 Wm^{-2}
Water temperature	°C	I-sensor, PT 100	±0.3%
Flow rate	LPM	KFCM-1000 K-2101083–2	±0.5%
Wind speed	ms^{-2}	Clima sensor, US	±5%
Wind direction	degree	Clima sensor, US	±5%
Ambient pressure	hPa	PTB-220TS, VAISALA	±5 hPa

The 3D model, which was drawn in the fixed north–south direction in Google Sketchup, was oriented 45° in the simulation studio using an equation solver named "orientation" in Fig. 6, in order to depict the exact orientation of the greenhouse. This orientation was linked to the equation solver named "azimuth", to define all the possible orientation for external walls and windows. Since the investigated greenhouse is located in South Korea, the setting of the hemisphere in the TRNbuild was fixed to the Southern and all other components necessary for the dynamic simulation are shown in Table 2. The ambient data needed for the simulation, solar radiation (SR), temperature, relative humidity, wind speed, and wind direction were saved in the weather data reader (Type 9). The observed SR was measured on a horizontal surface and the diffuse, beam, and reflected solar radiation was computed by linking Type 9 with the solar radiation processor (Type 16) using the Reindl et al. (1990) correlation in Eq (4)–(6):

$$\frac{I_d}{I} = 0.426kt - 0.256\sin(\alpha_s) + 0.00349T_{\text{amb}} + 0.0734\left(\frac{RH}{100}\right) \tag{4}$$

where I_d represents the diffuse radiation on the horizontal surface ($kJh^{-1}m^{-2}$), I denotes the total radiation on the horizontal surface ($kJh^{-1}m^{-2}$), kt represents the clearness index, α_s denotes the solar altitude angle (°), T_{amb} represents the ambient temperature (°C), and RH represents the relative humidity (%).

The horizontal beam radiation (I_b) is determined as the difference between the diffuse and the total radiation on the surface (Eq. (3)):

$$I_b = I - I_d \tag{5}$$

The total radiation on the tilted surface (I_T) is determined from Eq. (4):

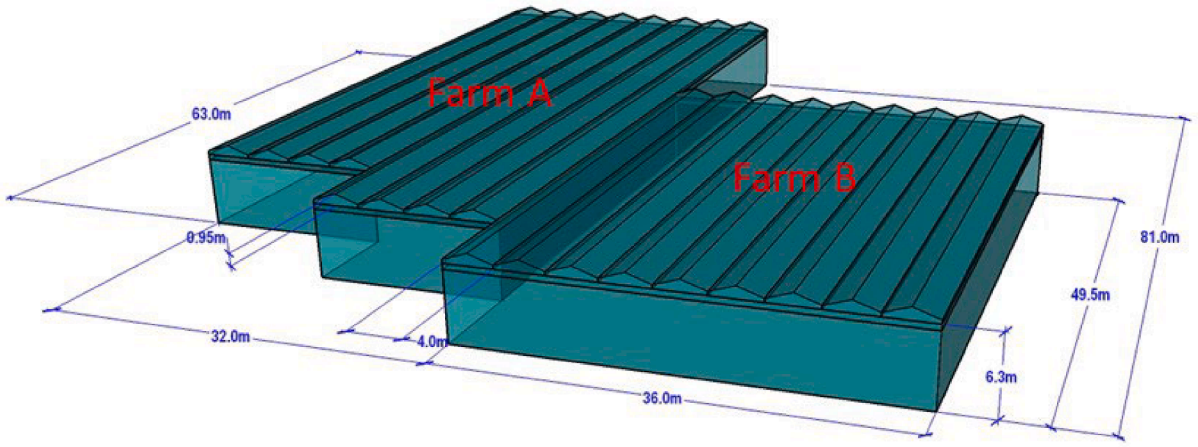


Fig. 5. 3D model of the greenhouse.

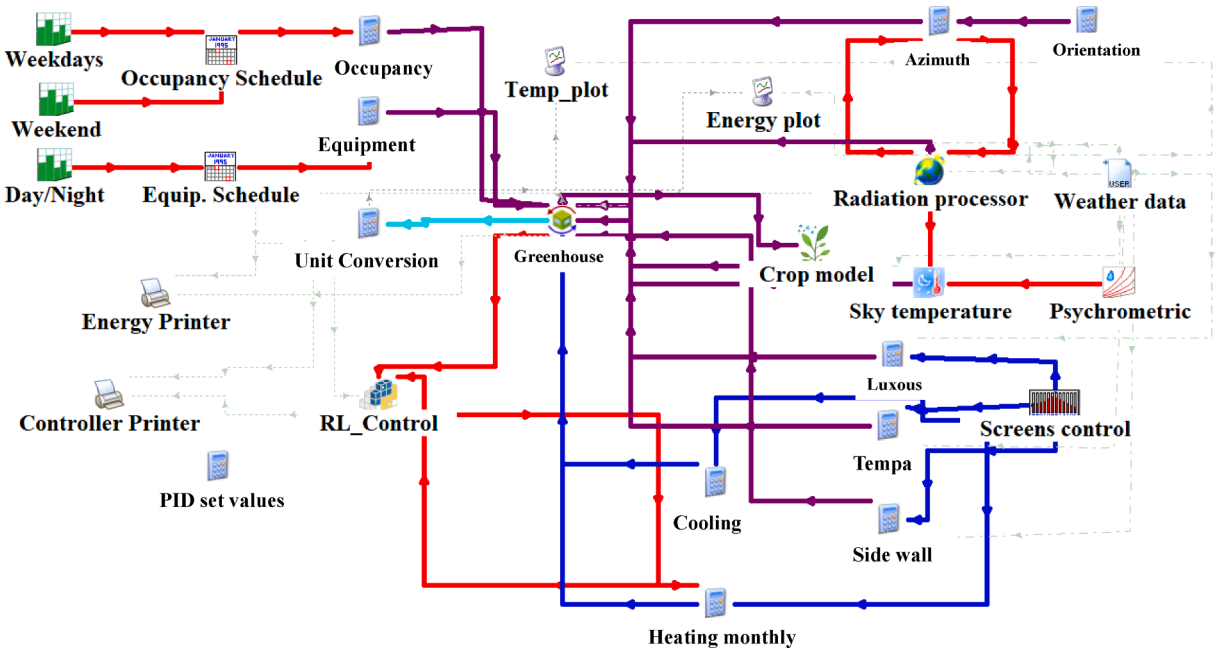


Fig. 6. TRNSYS simulation studio interface.

$$I_T = I_b \bullet \frac{\cos\theta}{\cos\theta_z} + I \bullet 0.5(1-\cos\beta)\rho_g + I_d \bullet 0.5(1+\cos\beta) \quad (6)$$

where θ represents the beam radiation incident angle (°), θ_z denotes the solar zenith angle (°), β is the surface slope (°), and ρ_g denotes the ground reflectance (−).

Further, Type 9 was linked with the psychrometric component (Type 33) to obtain data for the dry bulb temperature and relative humidity in order to calculate the dewpoint temperature. Both Type 16 and Type 33 were linked with the sky temperature component (Type 69) to determine the effective sky temperature for the calculation of the radiative heat transfer and the external surface longwave radiation exchange.

In the 3D model depicted in Fig. 5, the greenhouse was segmented into three zones, each delineated by thermal screens. Drawing an analogy to residential buildings, where shading devices adjust based on SR, the SR data from Type 9 was integrated with the monthly scheduler component (Type 518). As the greenhouse employed three distinct screens, each operating on a unique schedule, three equation solvers named "Luxuos," "Tempa," and "Obscura" were employed to govern the controls for these screens. Additionally, equation solvers labelled "heating monthly" were utilized to establish the requisite setpoint temperature within the greenhouse.

Meanwhile, people, equipment, and plants were all considered as the greenhouse loads that affect the thermal capacitance of the greenhouse. People and equipment scheduling were implemented using the Type 14 forcing function in the simulation studio. The ASHRAE130W-person gain type was selected for the occupancy gain, and a radiative and convective power of 0.5 kJ h^{-1} was selected as the equipment gain. Compared with the occupancy and equipment schedule, the plant density is significantly different during the growth and fruiting stages. Therefore, a stand-alone component (Type 780) that considers the transient change of the crop growth was used to estimate the heat flux induced by the crop. Type 780 estimates the crop evapotranspiration based on the step-by-step Penman–Monteith algorithm (Zotarelli et al., 2014). The evapotranspiration rate is estimated as the sum of the evapotranspiration from wind and radiation.

Evapotranspiration by wind is described by the following equation:

$$ET_{wind} = PT \times TT(e_s - e_a) \quad (7)$$

Table 2
Greenhouse components description in TRNSYS 18.

Component	Type	Description
Greenhouse	56	Analyzes greenhouse thermal behavior. It was also used to model the greenhouse ventilation.
Weather data	9	Reads the user-defined weather data.
Solar radiation processor	16	Use the total direct solar radiation on the horizontal surface, ambient temperature, RH, and building slope to calculate the total, beam, reflected, and diffuse radiation on all greenhouse tilt surfaces.
Psychrometer	33	Use the ambient temperature and the relative humidity are to calculate the dew point temperature.
Sky temperature	69	Calculate the sky temperature using the horizontal beam radiation, the horizontal diffuse radiation, the dry bulb temperature, and the dew point temperature as inputs.
Forcing function	14	Set the weekly equipment and occupancy schedule, and the daily temperature setpoint.
Forcing function sequencer	41	Identifies the unique day of the week.
Monthly schedule	518	Controls the opening and closing of the screen based on solar radiation.
Equation solver	–	To define the constant and necessary equations needed during the simulation.
Crop model	780	Computes the rate at which the vegetated surface vaporizes the readily available soil water using the Penman–Monteith algorithm.
Controller	661	Controls the natural ventilation and airflow through the thermal screens,
Controller	165	On/off controller for HVAC system
Controller	23	PID controllers for HVAC system
Plots and printer	65, 25	Displays and prints simulation results.
Python	3157	Calling Python script to control simulation steps and to exchange actions and observations with RL agent.

where,

$$PT = \frac{\gamma}{\Delta + \gamma(1 + C_d \times U_2)} \tag{8}$$

$$TT = \frac{C_n \times U_2}{T_k} \tag{9}$$

and,

$$\gamma = \frac{C_p \times P_{atm}}{h_{lg} \times \lambda} \tag{10}$$

where e_s denotes the average saturation pressure (kPa), e_a represents the vapor pressure of water (kPa), PT is the pressure term, Δ is the slope of the vapor pressure curve, C_d is a constant that the depends on the SR in the greenhouse and the crop type, TT is the temperature term, C_n is also dependent on the crop type but the value does not change between day and night unlike C_d, U_2 represents a constant used to estimate the wind velocity at the crop height (ms^{-1}), T_k denotes the mean temperature (°C), γ is the psychrometric constant, C_p is the specific heat of air ($J\ kg^{-1}\ K^{-1}$), P_{atm} represents the atmospheric pressure (kPa), h_{lg} is the latent heat of vaporization of water (J), and λ is the ratio of the molecular weight of water vapor to the molecular weight of dry air (–).

The evapotranspiration by radiation is also described by the following equation:

$$ET_{radiation} = DT \times R_{ng} \tag{11}$$

and,

$$DT = \frac{\Delta}{\Delta + \gamma(1 + C_d \times U_2)} \tag{12}$$

where R_{ng} represents the difference between the incoming shortwave radiation and the outgoing longwave radiation ($MJ\ m^{-2}h^{-1}$).

Specifically, we utilize the built-in HVAC system in TRNbuild, as the air-conditioning system. By default, this control is done internally in TRNbuild during energy rate control mode by setting the minimum comfort temperatures for the occupants, in this case, the crops in the greenhouse and the maximum heating/cooling power. TRNbuild assumes that the heating type manager uses an idealized heating control and does not account for hysteresis that could occur with feedback controllers. Alternatively in this paper, a more flexible external control was implemented using TRNSYS controllers integrated with equation solver. The external on/off, unoptimized PID, and optimized PID controller was implemented by routing the control output from the on/off controller (Type 165), PID controller (Type 23), and a self-developed PID-RL controller through the equation solver tagged “heating monthly” in the simulation studio before connecting it to TRNbuild. Additionally, the PID parameters were stored in the equation solver named “PID set values” in the simulation studio interface shown in Fig. 6. Apparently, this interface serves as the assembly panel to create, modify, and run the integrated model as an entity. Finally, by using an online plotter (Type 65) and a printer (Type 25), the greenhouse temperature, energy demand from TRNbuild, and the control signal from the PID-RL controller are visualized and printed to an external file for RL agent to act upon.

3.3.2. Self-tunning of PID parameters

The effectiveness of PID controller on the overall performance of a system relies heavily on how well it has been tuned. The performance is characterized by response time, overshoot, settling time and steady state error. The tuning of PID parameters is broadly grouped into classical and optimization techniques (Abbas & Mustafa, 2023). Aside from tuning by trial and error, the most common of the classical technique is the Ziegler – Nichols rule (Ziegler & Nichols, 1995). This rule elucidates guidelines for finding the gains for the P, I, and D terms based on the transient response characteristic of a specific model. In this work, the proposed DRL tuning strategy belongs to the optimization-based technique while the manual tuning technique adopted is the Ziegler – Nicholas method.

3.3.3. Formulating TRNSYS model as RL environment

TRNSYS offers a utility folder that facilitates the integration of external programs, such as CONTAM, EES, ESP-r, Excel, MATLAB, and Python. Python, distinct from the others, has emerged as the preferred language within the ML community, owing to renowned packages like scikit-learn, TensorFlow, and PyTorch that cater to Python users. TRNSYS standard components, specifically Type 163 and Type 169, enable the transfer of TRNSYS environment states to Python, facilitating the operation of RL agents. Type 163 utilizes both Fortran-based input/output files and Python, albeit with limited speed. In contrast, Type 169, coded in C++, leverages Python’s C-API principle for improved performance. While Type 169 embeds Python within the code, it necessitates extensive C wrapper code to enable direct variable communication between TRNSYS and Python. Alternatively, users employ a custom user defined TRNSYS component known as Type 3517, which facilitates data exchange with Python via the CFFI (C Foreign Function Interface) package. CFFI allows users to define an interface for predefined Python functions, generating a DLL (referred to as CFFI DLL) that interact with both the TRNSYS DLL and the Python DLL during simulation.

Generally, for an RL agent to get trained by most RL algorithms, an unlimited trial with access to full or partial states and control action on each timestep is essential. Usually, the dynamic model of the problem serves as the playground for the RL agent; in this case, our greenhouse model in TRNSYS. Theoretically, adding a means to reset the terminal states and step the model simulations to return the appropriate state after applying the supplied action at each timestep should suffice to formulate any dynamic model as an RL environment. A well-known framework in the community is the OpenAI gym (Brockman et al., 2016). Unfortunately, TRNSYS does not offer an out-of-the-box direct way to step simulating model manually (that is, it is designed to run an

uninterrupted simulation from start to end). Researchers have implemented various methods to enable RL agents' stepwise interaction with building simulation tools. For instance, Gao et al. (2019) implemented MATLAB–MySQL and Chen et al. (2023) used Python–filelock to synchronize simulation steps and exchange state–action information. Our approach follows the latter; however, we utilize two Python shells. One shell runs the implementation provided by TRNSYS Type 3157 (Bernier et al., 2022) to lock and write the observation file and block simulation until the next action is unlocked as shown in Algorithm 1. The other shell is an external (to TRNSYS) Python shell that wraps the environment as an OpenAI gym, named trnsys_env (that is TRNSYS-Environment). The trnsys_env is a class that inherits the OpenAI gym and provides methods to override the reset, step, and close methods in the parent class. The reset method terminates its previously instantiated TRNSYS process (if any) and starts another using the TRNSYS input file (deck file). The step method writes and unlocks the action file to step the simulation and reads the observation file then locks the action file to pause the simulation until the next action is available. The closed method is a means to terminate instantiated TRNSYS. Algorithm 2 illustrates the complete workflow in trnsys_env.

Algorithm 1: Implementation on the Type 3157

```
Require: Inputs: observation, outputs: action
Initialization (TRNData):
1:                          if is_busy
2:                          is_busy = False
3:                          if obs_is_locked
4:                          obs_is_locked = False
StartTime (TRNData):
1:                          obs = TRNData.inputs
2:                          with obs_is_locked
3:                          write (obsFile, obs)
Iteration (TRNData):
1:                          with is_busy
2:                          with obs_is_locked
3:                          write (obsFile, TRNData.inputs)
4:                          wait action == unlock
5:                          with act_is_locked
6:                          action = read (actFile)
7:                          TRNData.outputs = action
```

Algorithm 2: OpenAI Gym TRNSYS–Environment, trnsys_env

```
Require: templateDeckFile, config
step(action):
1:              with is_busy
2:              with actLock
3:              write (actFile, action)
4:              obs = read (obsFile)
5:              reward = compute_reward ()
6:              info = {}
7:              done = False (TRNSYS_PID OR Not EndOfSeason) else True
8:              return obs, reward, done, info
reset ():
1:              if TRNSYS_PID
2:              Task-kill TRNSYS_ProcID
3:              TRNSYS_ProcID = background-run trnsys.exe 'templateDeckFile'
4:              obs = read (obsFile)
5               return obs
close ():
1:              Task-kill TRNSYS_ProcID
2:              return
```

Like in MATLAB and Simulink, the RL state–action is formulated by deriving inspiration from the PID control, where the controller (v) aims at driving the error (e) between the setpoint and measured value to the minimum at any time (t). For a good tracking response, the P, I, and D gains are iteratively set in Eq. (13), conventionally, by trial and error. In TRNSYS, the implementation is as simple as using Type 23 with appropriately tuned parameters. According to TRNSYS's description of Type 23, the PID implementation follows a slight modification:

$$v(t) = K\left[e(t) + \frac{1}{T_i}\int_0^t e(\tau)\mathrm{d}\tau + T_\mathrm{d}\frac{\mathrm{d}e(t)}{\mathrm{d}t}\right] = P + I + D \tag{13}$$

However, Type 23 implementation of the PID controller places some restrictions on the nature of acceptable PID parameters, thus limiting the tuning freedom. The adoption and implementation of a discrete PID controller mitigated this constraint. Hence enabling the ground for the conventional tuning technique and the proposed DRL-based tuning technique described in subsection 3.3.4 to 3.3.7.

In this work, DRL is utilized in achieving optimal tuning of PID gains to not only optimize for minimal setpoint error but as well as to minimize energy consumption. This is contrary to the usual convention of using the trained RL agent itself. The reason for this path is because most greenhouse HVAC systems are already equipped with a PID controller, and the resource is only limited for its computation. Therefore, they do require additional computational overhead to inference RL agent.

3.3.4. Selecting observation for the RL environment.

To achieve this for our RL TRNSYS–environment shown in Fig. 7, the observation space is set as a (3 × 1) vector of $\left[e(t), \int_0^t e(\tau)\mathrm{d}\tau, \frac{\mathrm{d}e(t)}{\mathrm{d}t}\right]^T$, constituting the error term, the integral, and derivative of the error, respectively. The error is the difference between the setpoint and the measured temperature in Celsius (^{o}C). This observation is normalized around the possible minimum and maximum range observed during TRNSYS model validation. The error term ranges from –50 to + 50, the integral term ranges between –1,000 and + 1,000, and the differential term ranges from –1,000 to + 1,000. However, the observation are clipped in a situation where the values are out of these boundaries.

3.3.5. Setting the action for the RL environment

The action space is a 27 discrete space that maps to the possible combinations of [-1, 0, 1] for each of the P, I and D gains. The interpretation of this action is to determine whether to cause a negative, zero, or positive change to each gain by a specific predetermined delta value [Δ_p, Δ_i, Δ_d]. The resulting gains are then multiplied and summed up with the observation to compute the control efforts for the HVAC system as shown in Fig. 7. Shown in Table 3 is the list of the combinations and their respective index.

3.3.6. Formulating the reward function for the RL environment

The reward function was inspired by the MATLAB and Simulink. The reward function incorporates three components: an error term, an energy term, and a weight term for energy (w_{engy}). The error represents the disparity between the setpoint and the measured temperature in Celsius (°C), while the energy denotes the power consumed by the TRNbuild HVAC system during the simulation in kilowatt-hours (kWh). To ensure unit consistency of the energy and error term, the heat energy in Eq. (3) is utilized for conversion. This dimensionless weight term (w_{engy}) ranges between 0 and 1, allowing for the scaling of energy consumption to adjust the RL agent's priorities. Thus, the reward function is structured to minimize the sum of squared error and the weighted squared energy, as depicted in Eq. (14). Ultimately, the agent aims to maximize the cumulative reward, achievable through various RL algorithms proposed by researchers.

$$reward = -\left(error^2 + w_{engy} * energy^2\right) \tag{14}$$

By wrapping our TRNSYS model as an OpenAI gym environment, as presented in Algorithm 2, community implementations of RL algorithms automatically become available because many frameworks leverage the universality of the gym. This decoupling of RL algorithm and environment makes RL researchers often consolidate efforts to expand applications for more use cases or propose novel RL algorithms on existing applications. However, this work focuses on the former.

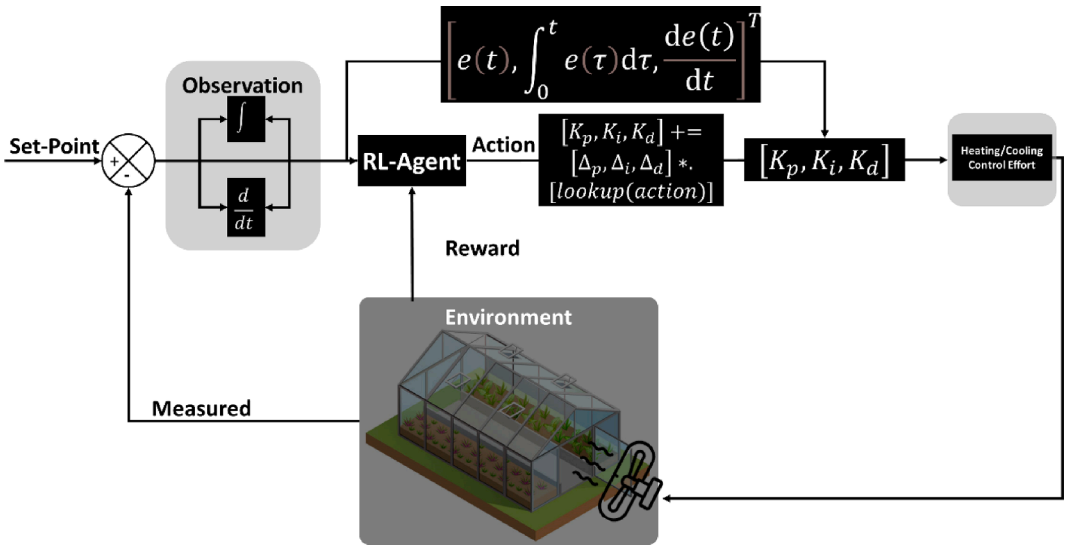


Fig. 7. Formulation for the RL environment.

Table 3
Mapping of the discrete action space to the possible combinations of [-1, 0, 1] for each of the P, I and D gains.

Index	Kp	Ki	Kd
0	−1	−1	−1
1	−1	−1	0
2	−1	−1	1
3	−1	0	−1
4	−1	0	0
5	−1	0	1
6	−1	1	−1
7	−1	1	0
8	−1	1	1
9	0	−1	−1
10	0	−1	0
11	0	−1	1
12	0	0	−1
13	***0***	***0***	***0***
14	0	0	1
15	0	1	−1
16	0	1	0
17	0	1	1
18	1	−1	−1
19	1	−1	0
20	1	−1	1
21	1	0	−1
22	1	0	0
23	1	0	1
24	1	1	−1
25	1	1	0
26	1	1	1

3.3.7. Adoption of the DQN algorithm

This work focuses on building RL environment to leverage RL algorithm's potential in solving the pressing issue in greenhouse. Therefore, rather than designing a new RL algorithm, we opt to select potential existing algorithm for the problem. We acknowledge that unique application problems that perform unsatisfactorily on existing algorithms have given birth to novel RL algorithms, thus limiting these RL algorithms to a specific scope. A typical example is the type of observations and actions they could handle. In this case, our observations and actions are continuous and discrete respectively, thus making the popular Q-Learning (Watkins and Dayan, 1992) and a few others not best suited for training our agents. Although Fujimoto et al. (2018) and Qiu et al. (2023) presented that TD3 and improved TD3 perform well for heating load energy optimization. However, our motive is to optimize P, I and D gains to minimize tracking error and heating energy with RL agent by taking discrete actions. Therefore, this work adopts the Deep Q-Network (DQN) algorithm in training the proposed RL agent (Mnih et al., 2013).

As mentioned earlier, DQN is a version of Q-Learning that utilizes neural network as the universal approximator instead lookup/transition table used in Q-Learning. The agent in this case is a multilayer perceptron (MLP) comprising of two fully connected hidden layers with 400 and 300 neurons and a rectified linear unit as the activation function. The agent accepts continuous input observations and returns a discrete action. The discrete action is translated to HVAC control effort by interpreting it as a combination to cause a negative (−1), zero (0), or positive (+1) change of delta to each gain. For instance, given that the agent outputs an action value of 21, the corresponding action lookup is [1, 0, −1] in Table 3. For a fixed delta change of 0.001, it means the current K_P should be incremented by 0.001, K_I should be incremented by 0 (remain unchanged) and K_D should be decremented by 0.001. The summation of the resulting new K_P multiplied by error, K_I multiplied by integral of error and K_D multiplied by differential of error gives the HVAC control effort.

However, during training, we implemented early stopping as an intervention to avoid meaningless exploration and to enhance the useability of the actions learned by the agent. As some of the gains learned by the agent might not be practically realistic and are found to easily saturate the idealized HVAC system in TRNbuild. Therefore, our stable controller learned to discard gains (restored last best gain) that are not yielding improved cumulative rewards when compared to the previously earned rewards (at the of planting duration). After several repeated discarding, the training is stopped, as such, the agent is able to avoid meaningless exploration and reduce the computation cost. Algorithm 3 shows the implementation of the PID-DQN algorithm.

Algorithm 3: PID-Deep Q-learning

Initialize replay buffer D to capacity N
Initialize action-value function Q with random weights
Initialize K_P, K_I, and K_D
Initialize $[\Delta_p, \Delta_i, \Delta_d]$ where $\Delta \in (0, 1)$
lookup(index) ← Table 3
Initialize best reward r_{best}
for episode = 1, M do
  Initialize sequence s_1 and episodic cumulative reward r_{cumu}
  for t = 1, T do
    With probability ϵ select a random action a_t
    otherwise select $a_t = \max_a Q^*(s_t, a; \theta)$

(continued on next page)

(*continued*)

Algorithm 3: PID-Deep Q-learning

$[K_P, K_I, K_D] = [K_P, K_I, K_D] + [\text{lookup}(a_t)]$ *. $[\Delta_p, \Delta_i, \Delta_d]$
$[e, \int, de/dt] = s_t$
Control-Effort = K_P * e + K_I * $\int$ + K_D * de/dt
Execute Control-Efforts in TRNSYS and observe reward r_t and state s_{t+1}
Update cumulative reward $r_{cumu} \mathrel{+}= r_t$
Store transition (s_t, a_t, r_t, s_{t+1}) in D
Sample random minibatch of transitions (s_j, a_j, r_j, s_{j+1}) from D

$$\text{Set } y_j = \begin{cases} r_j & \text{for terminal } s_{j+1} \\ r_j + \gamma \max_a Q(s_{j+1}, a; \theta) & \text{for non-terminal } s_{j+1} \end{cases}$$

Perform a gradient descent step on $(y_j - Q(s_j, a_j; \theta))^2$ according to equation (2)
end for
Apply gains discard criteria.

$$[K_P, K_I, K_D]_{lastBest} = \begin{cases} [K_P, K_I, K_D] & if\, r_{cumu} > r_{best} \\ [K_P, K_I, K_D]_{lastBest} & otherwise \end{cases}$$

end for

3.4. Statistical index for model evaluation

The TRNSYS model was validated against the experimental measurements using the Nash–Sutcliffe coefficient (NSE) and the root mean square error (RMSE). NSE quantifies the fit of the simulated data with the experimental data in a 1:1 plot. Its value ranges from $-\infty$ to 1, with values close to 1 indicating a substantial predictive potential of the model. The RMSE shows the magnitude of the error between the simulated and experimental measurements; the smaller the RMSE value, the better the model. The mathematical relationship of the validation indices is presented in Eqs. (15) and (16):

$$NSE = 1 - \left[\frac{\sum_{i=0}^{n}\left(y_i - \check{y}_i\right)^2}{\sum_{i=0}^{n}\left(y_i - \dot{y}\right)^2}\right] \tag{15}$$

$$RMSE = \frac{\sqrt{\sum_{i=0}^{n}\left(y_i - \check{y}_i\right)^2}}{n} \tag{16}$$

where N denotes the total number of observations, y_i represents the measured value, $\check{y}_i$ is the simulated value, $\dot{y}$ denotes the mean of the observed values, and n is the total number of observations.

4. Results

As the first step in the proposed optimization of PID parameters with DRL relies on a state-space model, specifically TRNSYS, we run a simulation of the greenhouse climate that is controlled by on/off regulation from October 2021 to March 2023. In contrast to residential buildings, which experience occupied and unoccupied periods with a fixed human comfort temperature, plants are consistently present in greenhouses with varying nighttime and daytime comfort temperatures. For tomatoes, the comfort temperature varies according to variety and region but is typically between 13 °C and 16 °C during the night and 23 °C and 27 °C during the day (Ogunlowo et al., 2021; Soussi et al., 2022). In the experimental greenhouse at Yeoju, the nighttime temperature is maintained between 15 °C and 20°C, while the daytime temperature is regulated within the range of 30 and 35°C, primarily due to the lower cooling capacity of 8.13 kW of the FCU compared to its heating capacity of 14.03 kW (Adesanya et al., 2024). These controlled setpoints serve as the threshold for temperature control during the simulation.

4.1. Climate conditions during the experimentation period

Fig. 8 illustrates the total monthly SR measured on a horizontal surface and the monthly average outside temperature throughout the experiment. From December 2021 to May 2022, the SR gradually increased from 89.8 kWh/m^2 to 205.7 kWh/m^2, declining to 163.6 kWh/m^2 in June 2022. A similar trend was observed in the second year, with a reduction in SR from December 2022 to February 2023. The hottest and coldest months occurred in May and December 2022, respectively, featuring mean monthly outside temperatures of 18.1 °C and −4.7°C, along with total horizontal SR of 205.7 kWh/m^2 and 82.3 kWh/m^2, respectively. During the first year of heating, the lowest and highest hourly temperatures, −17.8 °C and 20.2°C, were recorded on December 26, 2021, at 07:00, and March 12, 2022, at 15:00, respectively. In the second year, the lowest and highest hourly temperatures, −20.7 °C and 25.9°C, were recorded on January 26, 2023, at 07:30, and March 22, 2023, at 14:40, respectively.

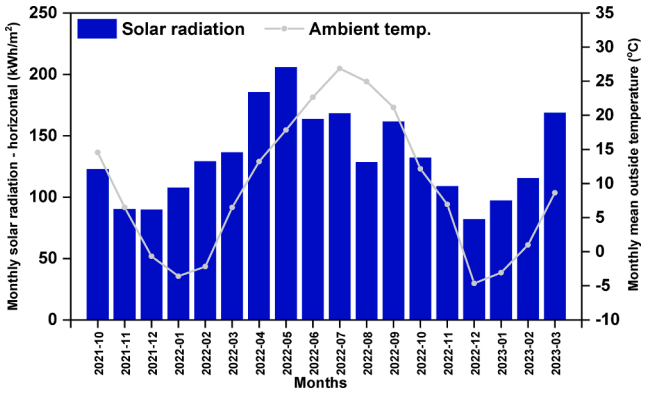


Fig. 8. Total monthly horizontal solar radiation and monthly average ambient temperature.

4.2. Greenhouse load in TRNSYS

The solar heat gained in the greenhouse significantly influences both microclimate conditions and energy consumption. TRNbuild employs various modes—simple, standard, and detailed—for calculating solar heat gain in the greenhouse, and the choice among them depends on the greenhouse's design. The TRNbuild standard and detailed modes, as validated in a prior study (Adesanya et al., 2022), accurately predicted the temperature in the concave and convex shapes of farms A and B. Consequently, these two modes were employed to assess the total greenhouse load in the TRNSYS simulation.

Fig. 9 presents a comparison between the total monthly simulated on/off control and experimental energy consumption. It is evident that

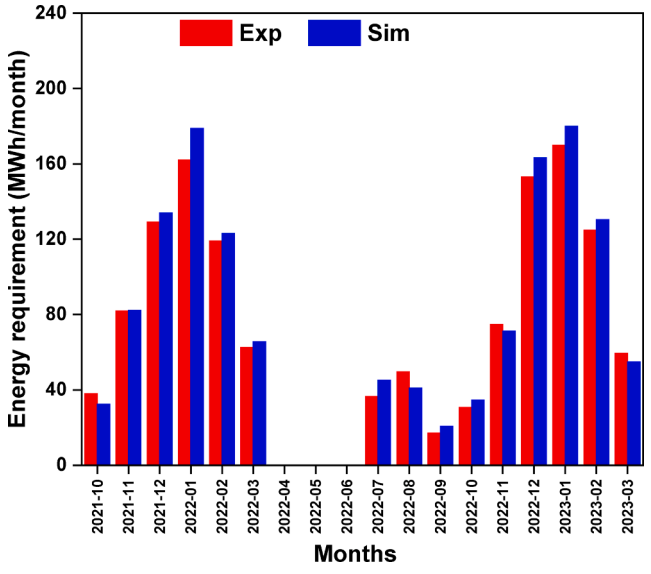


Fig. 9. Comparison of the energy requirements between experiment and simulation.

in some months, external factors contributed to the lower experimental energy consumption compared to the simulated energy demand. For instance, during the coldest periods, more heat is lost from the greenhouse, and the HVAC system supplies excess energy in order to maintain the greenhouse set-point temperature (Adesanya et al., 2023). The measured experimental energy consumption steadily increases by 0.35 %, 3.57 %, and 9.38 % compared to the simulated results as the average outside temperature decreases from 6.5 °C to −0.67 °C and − 3.59 °C in November 2021, December 2021, and January 2022, respectively. In February 2022, the deviation reduces to 3.26 % as the outside daily temperature increases by 1.41 °C compared to the January 2022 baseline. The total simulated energy demand resulted in 1357.50 MWh, while the experimental energy consumption was 1308.57 MWh during the investigation period. Overall, the calculated NSE and RMSE were 0.98 and 6.6 MWh, respectively. This suggests that the RL agent can rely on the validated on/off control model as its RL environment.

4.3. Energy efficiency of the manually tuned and optimized PID parameters

The validated on/off controller results obtained from the TRNSYS simulation served as the baseline for comparing the performance between the manually tuned PID parameters and the PID parameters optimized by the proposed DRL. Given the precision and accuracy of the Farm B detailed mode in predicting energy consumption compared to the Farm A standard mode, it was chosen as the OpenAI gym RL environment for optimizing PID parameters. The hardware environment for the cosimulation of TRNSYS and Python involved a notebook computer equipped with an Intel Core i7-6700 CPU @ 3.40 GHz, 16.0 GB of RAM, and a GTX 1050 GPU. For the sake of simplicity and ease of deployment, the cosimulation was conducted over a 59-day period, spanning from January to February 2022.

The RL agents underwent training using the Deep Q-Network, repeated four times to produce outcomes labelled RL-1, RL-2, RL-3, and RL-4, each corresponding to different PID optimized parameters tested in the simulation environment. The training was set to 50 thousand timesteps, and each timestep corresponds to approximately 1 h of simulation step. Therefore, 1 episode is approximately 1417 TRNSYS simulation timesteps. Other hyperparameters are presented in Table 4. It is also important to mention that early stopping was utilized to prevent input saturation for the controller, as it was observed that some optimal parameters are not realistic for TRNSYS to handle. To allow the trained DRL agent to use its optimized PID parameters to control the greenhouse during validation, the action is fixed at index 13 as shown in Table 3. This simply means all subsequent stepwise incremental changes to the optimized parameter should be frozen.

Table 5 summarizes energy consumption across various control scenarios and presents the weighted parameter values utilized in the proposed reward for each agent. The values of the weighted parameters of the RL agents were chosen to emphasize the degree of energy minimization over deviation from setpoint tracking. The reported results pertain to a two-month period during the heating season in Farm B, spanning from January 1 to February 28, 2022. Notably, manually tuning the PID controller led to a 7.2% reduction in energy consumption, while all RL agents achieved reductions ranging from 8.84% to 12.99%. This varying degree of energy consumption minimization can be attributed to the "w_{engy}", which is the weighting parameter proposed in the reward model to control the degree energy minimization. As envisaged, increasing this weighting parameter resulted in more reduced energy consumption, although, this is at the expense of the minimum average temperature which is slightly lower than RL-3 with zero " w_{engy}". Nevertheless, this is still approximately tolerable and better than the manually tuned PID, thus underscoring the impact of optimized PID parameters on energy consumption minimization.

Table 4
Hyperparameter settings for training the RL agent.

Parameter	Value
Learning Rate	0.001
Discounting factor	0.99
Optimizer	Adam
Batch Size	32
Episode	228
Exploration Epsilon [start, end, rate]	[0.1, 1.0, 0.05]
Delta values	[0.001, 0.001, 0.001]
w_{engy}	∈ (0, 1)
Replay Buffer	1,000,000

4.4. Comparative analysis of greenhouse temperature for different scenarios

Four days during the simulation was selected to illustrate the differences between nighttime and daytime temperatures for various control scenarios and to assess whether the setpoint temperature is maintained according to the control methods. It should be noted that only the heating setpoint is set in the TRNbuild heating regime type. The cooling setpoint, which, in this case, is the maximum comfortable crop temperature, is not set as the considered period is winter and no cooling is required. Nevertheless, the natural ventilation mode is utilized using the TRNFLOW model in TRNbuild (Ogunlowo et al., 2023).

Fig. 10 depicts the variation in the greenhouse's internal temperature during February 2022. Without activating the HVAC system (depicted with grey colour in Fig. 10a) between 12:00 on February 1 to 23:00 on February 4, the greenhouse inside temperature was uncomfortable for the crop due to the low outside temperature and SR. The greenhouse nighttime temperature dropped as low as −2.26 °C when the outside temperature and SR were −11.75 °C and 8.87 Wh/m^2, respectively. However, as SR increased, the greenhouse's internal temperature steadily rose during the day, reaching a peak value of 16.76 °C when the outside temperature and SR were 2.04 °C and 654.35 Wh/m^2. These scenarios indicate that the greenhouse inside temperature can achieve reasonable thermal comfort during the day at higher outside temperatures and SR, even without the operation of the HVAC system. This depends on the spectral transmissivity of the greenhouse covering materials and the thermal inertia of the greenhouse. In this case, the greenhouse was constructed with glass that allows over 90 % of the visible range of SR to enter the greenhouse while preventing the escape of infrared radiation.

By controlling the HVAC system with on/off control, the nighttime temperature of the greenhouse was maintained at the setpoint with little deviation, while the daytime temperature increased by a maximum of 4.01 °C compared with the unconditioned greenhouse (depicted with a blue colour in Fig. 10a). The increased temperature can be attributed to the continuous nighttime heating that occurred during the previous days in the greenhouse. During the same period, the manually tuned PID controller was also utilized (depicted with a green colour in Fig. 10a). It was observed that there was no difference in the daytime temperature of the on/off and PID controllers because the HVAC system was off, and the greenhouse inside temperature was within a comfortable range. However, a significant disparity is observed during the night.

Further, Fig. 10b shows the greenhouse inside air temperature with the proposed and on/off control methods. As seen, the greenhouse inside air temperature with all the RL runs substantiates the effect of the DRL algorithm on the PID parameters. The DRL-based optimized PID parameter control strategy prevents nighttime temperatures from significantly exceeding the minimum required temperature by intelligently regulating the temperature based on the crop's thermal comfort. The temperature inside the greenhouse fluctuates less at the setpoint compared to the unoptimized PID parameters. This was because the manually tuned PID is constrained to all its PID gains to be positively signed. In contrast, the RL agent was able to explore outside this constraint, thus learning a negative derivative gain consistently to

Table 5
Comparison of the energy consumption with different control scenarios.

Control method	PID parameters	Weighting parameters	Energy consumption (MWh)	Avg. nighttime temperature below setpoint (°C)	% Energy reduction
On/off	–	–	143.95	14.51	
PID	P = 0.9, I = 0.1, and D = 0.005	–	133.58	13.91	7.2
RL-1	P = 0.4252, I = 0.2097, and D = -0.1198	w_{engy} = 0.001	127.79	14.59	11.2
RL-2	P = 0.7426, I = 0.2542, and D = -0.069	w_{engy} = 0.005	125.25	14.53	12.99
RL-3	P = 0.7844, I = 0.1479, and D = -0.0081	w_{engy} = 0	131.21	14.69	8.8
RL-4	P = 0.7972, I = 0.161, and D = -0.0516	w_{engy} = 0.0001	130.61	14.68	9.3

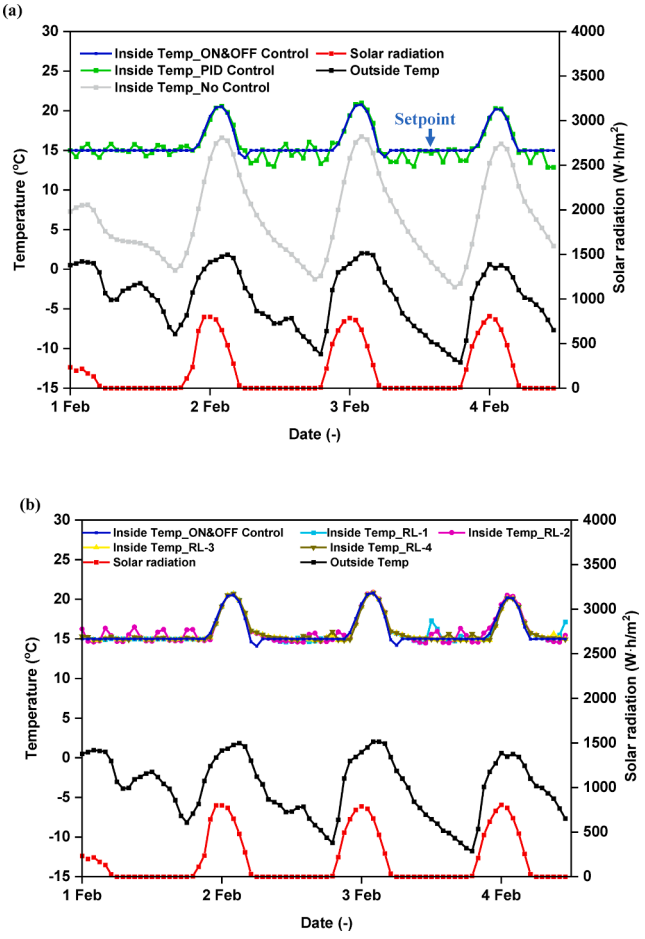


Fig. 10. Comparison of the greenhouse temperature with different controls (a) Uncontrolled, on/off, and manually tuned PID and (b) on/off, and optimized PID with DRL.

dampen the oscillation around the setpoint. Although having negative gains is unusual but not impossible. This possibly imply an inverse relationship between the error and the control action. In decreasing order, the average minimum observed temperature deviation from setpoint is 1.09°C, 0.47°C, 0.41°C, 0.32°C, and 0.31°C, for manually tuned PID, RL-2, RL-1, RL-4, and RL-3, respectively. As observed, RL-2 with the highest deviation of the RL from the setpoint achieved the highest energy consumption reduction of 12.99 %, indicating the influence of controlled temperature on energy consumption.

The apparent discrepancy in trends between the on/off and PID controls can be attributed to two factors. Firstly, our study employed an idealized heating system in TRNbuild, where the on/off controller automatically fixed the greenhouse temperature to the setpoint (just like a baseline), while our proposed PID controller dynamically adjusted the setpoint based on rewards from the RL-agent, prompting a more aggressive response aimed at swiftly minimizing the error between the setpoint and measured temperature in a short amount of time in order to reduce energy consumed. Secondly, the simulation time step of 1 h led to the aggregation of control decisions, resulting in a perceived stability in the on/off control compared to the PID. These factors collectively influenced the observed trends in temperature fluctuations. It is noteworthy that the manually tuned PID parameters exhibited significant deviations from the setpoint temperature, as depicted in Fig. 10a, in contrast to the optimized PID parameters obtained through RL that penalizes the setpoint deviation, as shown in Fig. 10b.

5. Discussion and future research

5.1. Insights and limitations

The model utilized in this study represents the third stage of improvement aimed at developing a robust integrated greenhouse model to enhance energy efficiency in the horticultural sector for farmers and stakeholders. Over the years, this model has undergone regular enhancements and has served as a resource for analysing the thermal performance of various greenhouse thermal screens (Rabiu et al., 2023) and assessing the impact of different renewable energy systems on greenhouse loads (Adesanya et al., 2023). The base model examined the effect of greenhouse structure on energy demand using the TRNSYS radiation mode (Adesanya et al., 2022), while the second improved model introduced a crop model based on the Penman-Monteith algorithm, investigating the influence of crops on heating and cooling loads (Adesanya et al., 2024). In its current form, the model incorporates a new control method using an idealized heating system in TRNbuild. This choice was made due to the absence of an FCU model in the TRNSYS library capable of operating in both heating and cooling modes based on manufacturer data (Calise et al., 2012). However, this study is significant because it addresses certain obstacles in TRNbuild, such as the fixation of building air temperature to the temperature setpoint during energy rate control and the constrained internal control of the idealized heating type manager.

Consequently, this present study evaluates the energy required to maintain crop thermal comfort during two heating seasons of greenhouse operation using three variants of heating systems in the multizone building model of TRNSYS software. While the greenhouse temperature fluctuates when the heating system is off (primarily during the day in the winter season), the first variant control (on/off) maintained the greenhouse temperature at the setpoint when the heating system was on (mostly during the night in the winter season). Ideally, it is not possible to maintain the greenhouse temperature precisely at the setpoint, but this setting in the heating type manager enables the analysis of the maximum amount of heat to be supplied to the greenhouse based on the control methods. In contrast, the other two variants of manually tuned and optimized PID control with RL caused the greenhouse temperature

to fluctuate around the setpoint when the heating system was on.

5.2. Future directions

In this study, we successfully tackled the challenges of obtaining optimal PID values from an RL agent, resulting in significant energy reduction compared to on/off and manually tuned PID controllers. The proposed methodology was tested in a virtual environment, with experimental energy consumption and observed greenhouse temperature serving as the baseline for comparison. Uncertainties recorded during the experiment were not accounted for in the virtual environment. However, future research could explore the feasibility of implementing the proposed method in a real greenhouse HVAC system with its negatively optimized PID parameters.

Furthermore, in future endeavours, the developed building model will be operated in TRNbuild temperature level control and connected to a real HVAC system to provide hot air, utilizing heat pumps as the heat source. This will demonstrate whether the energy savings achieved by the proposed control method can be replicated across different seasons of the year. To facilitate this, a new TRNSYS FCU model will be developed based on data files containing air dry bulb temperature, air wet bulb temperature, fluid mass flow rate, and inlet fluid temperature, as the introduced hot air conditions may deviate from the greenhouse setpoint temperature.

6. Conclusion

In this study, a combination of numerical and ML models was employed to simulate a greenhouse HVAC system. The developed methodology was tested on an ideal heating system in TRNbuild using on/off and PID controllers. The PID controller parameters were manually tuned and intelligently optimized using Ziegler-Nichols and DRL through TRNSYS-Python cosimulation. Unlike the on/off and manually tuned PI controllers, which lack an optimality criterion, the PID optimized controller from RL algorithm follows a reward function to improve the performance of the HVAC system at each simulation time step. The RL agents were trained on the Deep Q-Network, with training repeated four times, resulting in corresponding results tagged RL-1, RL-2, RL-3, and RL-4. The performance of the optimized PID parameter control was compared with that of simulated on/off and manually tuned PID parameter controllers in terms of crop thermal comfort and energy-saving potential. The crop thermal comfort temperature with manually tuned PID control, RL-1, RL-2, RL-3, and RL-4, was reduced by 1.09°C, 0.41°C, 0.47°C, 0.31°C, and 0.32°C, respectively, leading to a 7.17 %, 11.19 %, 12.99 %, 8.81 % and 9.23 % reduction in energy consumption compared with the on/off baseline control. This study has established a framework for enhancing the energy efficiency of greenhouse HVAC system operations.

This study was funded by the Korea Institute of Planning and Evaluation for Technology in Food, Agriculture, Forestry, and Fisheries (IPET) through the Agriculture, Food, and Rural Affairs Convergence Technologies Program for Educating Creative Global Leaders, funded by the Ministry of Agriculture, Food and Rural Affairs (MAFRA) (717001–7). This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2019R1I1A3A01051739). This work was supported by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry (IPET) through Agricultural Energy Self-Sufficient Industrial Model Development Program, funded by Ministry of Agriculture, Food and Rural Affairs (MAFRA) (120096–3).

CRediT authorship contribution statement

Misbaudeen Aderemi Adesanya: Conceptualization, Writing – original draft, Software, Methodology, Formal analysis, Validation. **Hammed Obasekore:** Conceptualization, Writing – original draft, Software, Formal analysis, Validation. **Anis Rabiu:** Data curation, Writing – original draft, Formal analysis. **Wook-Ho Na:** Visualization, Investigation. **Qazeem Opeyemi Ogunlowo:** Data curation, Formal analysis, Writing – review & editing. **Timothy Denen Akpenpuun:** Formal analysis, Writing – review & editing. **Min-Hwi Kim:** Conceptualization, Methodology, Resources, Funding acquisition. **Hyeon-Tae Kim:** Investigation, Supervision, Writing – review & editing. **Bo-Yeong Kang:** Investigation, Supervision, Writing – review & editing. **Hyun-Woo Lee:** Investigation, Supervision, Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

References

Abbas, I. A., & Mustafa, K. (2023). A review of adaptive tuning of PID-controller: Optimization techniques and applications. *Int. J. Nonlinear Anal. Appl. In Press, 6822* (January), 2008–6822. https://doi.org/10.22075/ijnaa.2023.21415.4024

Adesanya, M. A., Na, W., Rabiu, A., Ogunlowo, Q. O., Akpenpuun, T. D., Rasheed, A., Yoon, Y.-C., & Lee, H.-W. (2022). TRNSYS simulation and experimental validation of internal temperature and heating demand in a glass greenhouse. *Sustainability, 14* (14), 8283. https://doi.org/10.3390/su14148283

Adesanya, M. A., Na, W. H., Kim, M. H., & L. h. w.. (2024). Energy-Economic-Environmental Analysis of a Net-zero Energy Greenhouse with Fan-coil units and Hot-water pipes: Experiment and Modelling. *Indoor and Built Environment*. https://doi.org/10.1177/1420326X241246075

Adesanya, M. A., Yakub, A. O., Rabiu, A., Owolabi, A. B., Ogunlowo, Q. O., Yahaya, A., Na, W.-H., Kim, M.-H., Kim, H.-T., & Lee, H.-W. (2023). Dynamic modeling and techno-economic assessment of hybrid renewable energy and thermal storage systems for a net-zero energy greenhouse in South Korea. *Clean Technologies and Environmental Policy*. https://doi.org/10.1007/s10098-023-02656-3

Afram, A., Janabi-Sharifi, F., Fung, A. S., & Raahemifar, K. (2017). Artificial neural network (ANN) based model predictive control (MPC) and optimization of HVAC systems: A state of the art review and case study of a residential HVAC system. *Energy and Buildings, 141*, 96–113. https://doi.org/10.1016/J.ENBUILD.2017.02.012

Ajagekar, A., Mattson, N. S., & You, F. (2022). Energy-efficient AI-based Control of Semi-closed Greenhouses Leveraging Robust Optimization in Deep Reinforcement Learning. *Advances in Applied Energy, 9*, Article 100119. https://doi.org/10.1016/j.adapen.2022.100119

Ajagekar, A., & You, F. (2022). Deep reinforcement learning based automatic control in semi-closed greenhouse systems. *IFAC-PapersOnLine, 55*(7), 406–411. https://doi.org/10.1016/j.ifacol.2022.07.477

Asad Rizvi, S. A., Pertzborn, A. J., & Lin, Z. (2023). Development of a Bias Compensating Q-Learning Controller for a Multi-Zone HVAC Facility. *IEEE/CAA Journal of Automatica Sinica, 10*(8), 1704–1715. https://doi.org/10.1109/JAS.2023.123624

Azuatalam, D., Lee, W. L., de Nijs, F., & Liebman, A. (2020). Reinforcement learning for whole-building HVAC control and demand response. *Energy and AI, 2*, Article 100020. https://doi.org/10.1016/j.egyai.2020.100020

Baudoin, W., Nono-Womdim, R., Lutaladio, N., Hodder, A., Castilla, N., Leonardi, C., Pascale, S. D., & Qaryouti, M. (2013). In *Cultural Practices and Environment* (pp. 42–55). CRC Press. https://doi.org/10.1201/b13737-8.

Bernier, N., Marcotte, B., & Kummert, M. (2022). *Calling Python from TRNSYS with CFFI*. https://doi.org/10.5281/ZENODO.6523104.

Biemann, M., Scheller, F., Liu, X., & Huang, L. (2021). Experimental evaluation of model-free reinforcement learning algorithms for continuous HVAC control. *Applied Energy, 298*, Article 117164. https://doi.org/10.1016/J.APENERGY.2021.117164

Brandi, S., Piscitelli, M. S., Martellacci, M., & Capozzoli, A. (2020). Deep reinforcement learning to optimise indoor temperature control and heating energy consumption in buildings. *Energy and Buildings, 224*, Article 110225. https://doi.org/10.1016/J.ENBUILD.2020.110225

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). *OpenAI Gym*. http://arxiv.org/abs/1606.01540.

Calise, F., Ferruzzi, G., & Vanoli, L. (2012). Transient simulation of polygeneration systems based on PEM fuel cells and solar heating and cooling technologies. *Energy, 41*(1), 18–30. https://doi.org/10.1016/j.energy.2011.05.027

Chen, C., An, J., Wang, C., Duan, X., Lu, S., Che, H., Qi, M., & Yan, D. (2023). Deep Reinforcement Learning-Based Joint Optimization Control of Indoor Temperature and Relative Humidity in Office Buildings. *Buildings 2023, Vol. 13, Page 438*, *13*(2), 438. https://doi.org/10.3390/BUILDINGS13020438.

Copot, C., Thi, T. M., & Ionescu, C. (2018). PID based Particle Swarm Optimization in Offices Light Control. *IFAC-PapersOnLine, 51*(4), 382–387. https://doi.org/10.1016/J.IFACOL.2018.06.096

Drgoňa, J., Arroyo, J., Cupeiro Figueroa, I., Blum, D., Arendt, K., Kim, D., Ollé, E. P., Oravec, J., Wetter, M., Vrabie, D. L., & Helsen, L. (2020). All you need to know about model predictive control for buildings. *Annual Reviews in Control, 50*, 190–232. https://doi.org/10.1016/j.arcontrol.2020.09.001

Du, Y., Li, F., Munk, J., Kurte, K., Kotevska, O., Amasyali, K., & Zandi, H. (2021). Multi-task deep reinforcement learning for intelligent multi-zone residential HVAC control. *Electric Power Systems Research, 192*, Article 106959. https://doi.org/10.1016/J.EPSR.2020.106959

Du, Y., Zandi, H., Kotevska, O., Kurte, K., Munk, J., Amasyali, K., Mckee, E., & Li, F. (2021). Intelligent multi-zone residential HVAC control strategy based on deep reinforcement learning. *Applied Energy, 281*, Article 116117. https://doi.org/10.1016/J.APENERGY.2020.116117

Durand-Estebe, B., Le Bot, C., Mancos, J. N., & Arquis, E. (2013). Data center optimization using PID regulation in CFD simulations. *Energy and Buildings, 66*, 154–164. https://doi.org/10.1016/J.ENBUILD.2013.07.053

Fujimoto, S., Van Hoof, H., & Meger, D. (2018). Addressing Function Approximation Error in Actor-Critic Methods. *35th International Conference on Machine Learning, ICML 2018*, *4*, 2587–2601. https://arxiv.org/abs/1802.09477v3.

Gao, C., & Wang, D. (2023). Comparative study of model-based and model-free reinforcement learning control performance in HVAC systems. *Journal of Building Engineering, 74*(May), Article 106852. https://doi.org/10.1016/j.jobe.2023.106852

Gao, G., Li, J., & Wen, Y. (2019). *Energy-Efficient Thermal Comfort Control in Smart Buildings via Deep Reinforcement Learning*. http://arxiv.org/abs/1901.04693.

Gao, G., Li, J., & Wen, Y. (2020). DeepComfort: energy-efficient thermal comfort control in buildings via reinforcement learning. *IEEE Internet of Things Journal, 7*(9), 8472–8484. https://doi.org/10.1109/JIOT.2020.2992117

Gao, P., Lu, M., Yang, Y., Wu, H., Mao, H., & Hu, J. (2024). Greenhouse light and CO2 regulation considering cost and photosynthesis rate using i-nsGA II. In. *Expert Systems with Applications, 237*. https://doi.org/10.1016/j.eswa.2023.121680

Ghaddar, D., Itani, M., Ghaddar, N., Ghali, K., & Zeaiter, J. (2021). Model-based adaptive controller for personalized ventilation and thermal comfort in naturally ventilated spaces. *Building Simulation, 14*(6), 1757–1771. https://doi.org/10.1007/S12273-021-0783-X/METRICS

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., & Silver, D. (2018). Rainbow: Combining improvements in deep reinforcement learning. In *32nd AAAI Conference on Artificial Intelligence*. https://doi.org/10.1609/aaai.v32i1.11796

Hu, G., & You, F. (2022). Model predictive control for greenhouse condition adjustment and crop production prediction. In *Computer Aided Chemical Engineering* (Vol. 51). Elsevier Masson SAS. https://doi.org/10.1016/B978-0-323-95879-0.50176-4.

Jeon, H., Kim, D. W., & Kang, B. Y. (2024). Deep reinforcement learning for cooperative robots based on adaptive sentiment feedback. *Expert Systems with Applications*, *243* (August 2023), 121198. https://doi.org/10.1016/j.eswa.2023.121198.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A. A., Yogamani, S., & Perez, P. (2022). Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems, 23*(6), 4909–4926. https://doi.org/10.1109/TITS.2021.3054625

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). *December 19)*. NIPS: Playing Atari with Deep Reinforcement Learning. https://arxiv.org/abs/1312.5602v1.

Norvig, S. J. R. and P. (2010). Artificial Intelligence: A Modern Approach. In S. J. R. and P. Norvig (Ed.), *Prentice Hall Series* (Third edit). Pearson. www.pearsonhighered.com.

Ogunlowo, Q. O., Akpenpuun, T. D., Na, W. H., Rabiu, A., Adesanya, M. A., Addae, K. S., Kim, H. T., & Lee, H. W. (2021). Analysis of heat and mass distribution in a single- and multi-span greenhouse microclimate. *Agriculture (Switzerland), 11*(9), 1–24. https://doi.org/10.3390/agriculture11090891

Ogunlowo, Q. O., Na, W. H., Rabiu, A., Adesanya, M. A., Akpenpuun, T. D., Kim, H. T., & Lee, H. W. (2022). Effect of envelope characteristics on the accuracy of discretised greenhouse model in TRNSYS. *Journal of Agricultural Engineering, 53*(3). https://doi.org/10.4081/jae.2022.1420

Opeyemi Ogunlowo, Q., Denen Akpenpuun, T., Ho Na, W., Aderemi Adesanya, M., Rabiu, A., Dutta, P., Kim, H., & Lee, H. (2023). Simulation of greenhouse energy and strawberry (Seolhyang sp.) yield using TRNSYS DVBES: A base case. *Solar Energy, 266*(August), Article 112196. https://doi.org/10.1016/j.solener.2023.112196

Prieto, J., Ajnannadhif, R. M., Fernández-del Olmo, P., & Coronas, A. (2023). Integration of a heating and cooling system driven by solar thermal energy and biomass for a greenhouse in Mediterranean climates. *Applied Thermal Engineering, 221*(December 2022). https://doi.org/10.1016/j.applthermaleng.2022.119928

Qin, H., Yu, Z., Li, T., Liu, X., & Li, L. (2023). Energy-efficient heating control for nearly zero energy residential buildings with deep reinforcement learning. *Energy, 264* (November 2022). https://doi.org/10.1016/j.energy.2022.126209

Qiu, Y., Zhou, S., Xia, D., Gu, W., Sun, K., Han, G., Zhang, K., & Lv, H. (2023). Local integrated energy system operational optimization considering multi-type uncertainties: A reinforcement learning approach based on improved TD3 algorithm. *IET Renewable Power Generation*. https://doi.org/10.1049/RPG2.12725

Rabiu, A., Adesanya, M. A., Na, W. H., Ogunlowo, Q. O., Akpenpuun, T. D., Kim, H. T., & Lee, H. W. (2023). Thermal performance and energy cost of Korean multispan greenhouse energy-saving screens. *Energy, 285*(September), Article 129514. https://doi.org/10.1016/j.energy.2023.129514

Rabiu, A., Na, W., Denen, T., Rasheed, A., & Aderemi, M. (2022). Determination of overall heat transfer coefficient for greenhouse energy-saving screen using Trnsys and hotbox. *Biosystems Engineering, 217*, 83–101. https://doi.org/10.1016/j.biosystemseng.2022.03.002

Rasheed, A., Kim, H. T., & Lee, H. W. (2022). Modeling-based energy performance assessment and validation of air-to-water heat pump system integrated with multi-span greenhouse on cooling mode. *Agronomy, 12*(6), 1374. https://doi.org/10.3390/agronomy12061374

Reindl, D. T., Beckman, W. A., & Duffie, J. A. (1990). Diffuse fraction correlations. *Solar Energy, 45*(1), 1–7. https://doi.org/10.1016/0038-092X(90)90060-P

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2015). Prioritized experience replay. *CoRR.*. https://doi.org/10.48550/arXiv.1511.05952

Shin, M., Kim, S., Kim, Y., Song, A., Kim, Y., & Kim, H. Y. (2024). Development of an HVAC system control method using weather forecasting data with deep reinforcement learning algorithms. *Building and Environment, 248*(June 2023), Article 111069. https://doi.org/10.1016/j.buildenv.2023.111069

Solinas, F. M., Macii, A., Patti, E., & Bottaccioli, L. (2024). An online reinforcement learning approach for HVAC control. *Expert Systems with Applications, 238*(PA), Article 121749. https://doi.org/10.1016/j.eswa.2023.121749

Soussi, M., Chaibi, M. T., Buchholz, M., & Saghrouni, Z. (2022). Comprehensive review on climate control and cooling systems in greenhouses under hot and arid conditions. *Agronomy, 12*(3). https://doi.org/10.3390/agronomy12030626

Soyguder, S., Karakose, M., & Alli, H. (2009). Design and simulation of self-tuning PID-type fuzzy adaptive control for an expert HVAC system. *Expert Systems with Applications, 36*(3), 4566–4573. https://doi.org/10.1016/J.ESWA.2008.05.031

Tune PI Controller Using Reinforcement Learning - MATLAB & Simulink. (n.d.). Retrieved May 2, 2023, from https://www.mathworks.com/help/reinforcement-learning/ug/tune-pi-controller-using-td3.html.

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-Learning. *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, 2094–2100.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., & De Freitas, N. (2016). Dueling Network Architectures for Deep Reinforcement Learning. *33rd International Conference on Machine Learning, ICML 2016*, *4*(9), 2939–2947.

Watkins, C. J. C. H., & Dayan, P. (1992). *Q-Learning* (Vol. 8).

Żabnieńska-Góra, A., Khordehgah, N., & Jouhara, H. (2021). Annual performance analysis of the PV/T system for the heat demand of a low-energy single-family building. *Renewable Energy, 163*, 1923–1931. https://doi.org/10.1016/j.renene.2020.10.123

Zhang, Z., Ma, C., & Zhu, R. (2016). Self-tuning fully-connected PID neural network system for distributed temperature sensing and control of instrument with multi-modules. *Sensors, 16*(10), 1709. https://doi.org/10.3390/s16101709

Ziegler, J. G., & Nichols, N. B. (1995). Optimum settings for automatic controllers. *InTech, 42*(6), 94–100. https://doi.org/10.1115/1.4019264

Zotarelli, L., Dukes, M. D., Romero, C. C., Migliaccio, K. W., & Morgan, K. T. (2014). Step by Step Calculation of the Penman-Monteith Evapotranspiration (FAO-56 Method). AE459. *Institute of Food and Agricultural Sciences. University of Florida*, 1–14. http%22//edis.ifas.ufl.edu/ae459.