Delft University of Technology

RL-Guided MPC for Autonomous Greenhouse Control

Msaad, Salim; Harraway, Murray; Mcallister, Robert D.

DOI
10.1016/j.ifacol.2025.11.829

Licence
CC BY-NC-ND

Publication date
2025

Document Version
Final published version

Published in
IFAC-PapersOnline

Citation (APA)
Msaad, S., Harraway, M., & Mcallister, R. D. (2025). RL-Guided MPC for Autonomous Greenhouse Control. *IFAC-PapersOnline*, *59*(23), 449-454. https://doi.org/10.1016/j.ifacol.2025.11.829

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 59-23 (2025) 449–454

RL-Guided MPC for Autonomous Greenhouse Control

Salim Msaad[*] Murray Harraway[*] Robert D. McAllister[*]

[*] *Delft Center for Systems and Control (DCSC), Delft University of Technology, Delft, The Netherlands*

Abstract: The efficient operation of greenhouses is essential for enhancing crop yield while minimizing energy costs. This paper investigates a control strategy that integrates Reinforcement Learning (RL) and Model Predictive Control (MPC) to optimize economic benefits in autonomous greenhouses. Previous research has explored the use of RL and MPC for greenhouse control individually, or by using MPC as the function approximator for the RL agent. This study introduces the RL-Guided MPC framework, where a RL policy is trained and then used to construct a terminal cost and terminal region constraint for the MPC optimization problem. This approach leverages the ability to handle uncertainties of RL with MPC's online optimization to improve overall control performance. The RL-Guided MPC framework is compared with both MPC and RL via numerical simulations. Two scenarios are considered: a deterministic environment and an uncertain environment. Simulation results demonstrate that, in both environments, RL-Guided MPC outperforms both RL and MPC with shorter prediction horizons.

Copyright © 2025 The Authors. This is an open access article under the CC BY-NC-ND license (https://creativecommons.org/licenses/by-nc-nd/4.0/)

Keywords: Greenhouse Control, Model Predictive Control, Reinforcement Learning, Economic Optimization

1. INTRODUCTION

The sustainable and efficient operation of greenhouses is pivotal to modern agriculture, offering a controlled environment to maximize crop yield while optimizing resource efficiency. Modern greenhouse systems demand advanced control strategies to dynamically manage climatic variables, such as temperature, humidity, and CO2 levels, with fluctuating external conditions. Traditional control methods struggle to balance long-term economic objectives with real-time operational demands. Reinforcement Learning (RL) and Model Predictive Control (MPC) have shown promise in addressing these challenges. Prior research has explored RL and MPC as standalone solutions for greenhouse automation. RL excels in handling stochastic environments, while MPC leverages model-based optimization to enforce constraints and recalibrate actions in real time. RL approaches for greenhouse control have been studied in van Laatum et al. (2024) and in Morcego et al. (2023). The first study presents an open-source RL environment for greenhouse control, comparing different RL approaches, while the second study introduces a RL-based controller that utilizes deep deterministic policy gradient for greenhouse climate control, comparing it to a MPC controller. Boersma et al. (2022) propose a robust MPC controller for greenhouse climate control, showing improved performance over traditional MPC. Mallick et al. (2025) integrate RL and MPC by using MPC as a function approximator within the RL framework.

The autonomous greenhouse control problem is an economic optimization task. Conventional MPC approaches focus on tracking an economically optimal steady-state for the system. However, for applications such as greenhouse control, steady-state tracking is not desirable as optimal economic operation requires dynamically adjusting the temperature, humidity, and CO2 levels in the greenhouse. Economic MPC, which uses economic costs directly in the stage cost and does not focus on stabilizing a steady-state target, is therefore more appropriate for greenhouse control. Nonetheless, a steady-state or reference trajectory is often used to improve and guarantee performance of economic MPC implementations (Amrit et al., 2011; Risbeck and Rawlings, 2020; Angeli et al., 2012)

This paper presents the RL-Guided MPC framework, where an RL policy informs the terminal cost and region constraints in the MPC optimization. The policy is used to build a cost function approximator for the terminal cost, while rollouts using the actor define the terminal region constraint. This integration improves control performance while ensuring computational efficiency in scenarios with limited prediction horizons. The efficacy of the proposed approach is validated through numerical simulations, comparing RL-Guided MPC against standalone MPC and RL controllers. The simulations are conducted in two environments. One environment is deterministic, while the other is stochastic with parametric uncertainty.

Section 2 outlines the greenhouse model, simulation environments, and the optimization problem tackled by the RL agent, MPC, and RL-Guided MPC. Section 3 and Section 4 describe the RL agent and MPC controller, while Section 5 introduces the RL-Guided MPC framework. Numerical results are presented in Section 6, and the conclusions are presented in Section 7.

Table 1. Physical meaning of input u, state x, and disturbance d

u_1	CO_2 injection ($\mathrm{mg\,m^{-2}\,s^{-1}}$)	x_1	dry weight ($\mathrm{kg\,m^{-2}}$)	d_1	solar radiation ($\mathrm{W\,m^{-2}}$)
u_2	ventilation ($\mathrm{mm\,s^{-1}}$)	x_2	indoor CO_2 (ppm)	d_2	outdoor CO_2 ($\mathrm{kg\,m^{-3}}$)
u_3	heating ($\mathrm{W\,m^{-2}}$)	x_3	indoor temperature (°C)	d_3	outdoor temperature (°C)
		x_4	indoor humidity (%)	d_4	outdoor humidity ($\mathrm{kg\,m^{-3}}$)

2. BACKGROUND

2.1 Greenhouse Model

The greenhouse and crop model utilized in this paper is the same as in Van Henten (1994). The model is discretised with the fourth order Runge-Kutta method with a sample period $\Delta t = 1800$ s (30 minutes), resulting in the following state space model:

$$x(k+1) = f(x(k), u(k), d(k), p) \tag{1}$$

with discrete time $k \in \mathbb{Z}^+$, state variable $x(k) \in \mathbb{R}^4$, control input $u(k) \in \mathbb{R}^3$ and weather disturbance $d(k) \in \mathbb{R}^4$. The parameter $p \in \mathbb{R}^{22}$ represents all parameters used in the model. The values and meaning of p and the nonlinear function $f(\cdot)$ are described in Boersma et al. (2022). State estimation is not considered in this work. It is assumed that the state variable $x(k)$ is always known. Table 1 provides the physical meaning of the state, input, and disturbance variables.

The weather data for simulations and training was sourced from the Venlow Greenhouse in Bleiswijk, covering the period from January 30 to March 11, 2014. The weather is assumed to be deterministic and known at each time step. This dataset spans a 40-day growing period with a time step of 30 minutes, resulting in a total of 1920 time steps. For the numerical simulations, two scenarios are considered: a deterministic environment and a stochastic environment. In a real-world greenhouse, uncertainty is present in all aspects of the model. Uncertainty may arise in the weather prediction, measurement noise on the outputs, and the control inputs may not be exact. As done in Boersma et al. (2022), the uncertainty in the stochastic environment is modelled as parametric uncertainty, which aims to offer a simplified representation of the real-world uncertainties. It is assumed that the uncertain parameters $\hat{p} \in \mathbb{R}^{22}$ follow the uniform probability distribution

$$\hat{p} \sim \mathcal{U}(p(1-\delta), p(1+\delta)), \tag{2}$$

where vector p represents the nominal values of the parameters and δ is a percentage that defines the range of the distribution. The value of δ is set to 5% for all of the stochastic simulations presented in Section 6. For each simulation in the stochastic environment, a new set of uncertain parameters $\hat{p}$ is sampled from the distribution in (2), and the system dynamics in (1), with $p = \hat{p}$, are used for simulation.

2.2 Optimization Problem

Optimal greenhouse control involves maintaining suitable environmental conditions for crop growth, including temperature, humidity, and CO_2 levels, while minimizing resource use. This is achieved through control inputs for heating using heating pipes, ventilation by controlling window openings, and CO_2 injection. The goal is to maximize crop yield while minimizing resource consumption.

Table 2. Pricing and penalty factors

Symbol	Value	Unit
c_1	$1.906 \cdot 10^{-1}$	€$\mathrm{mg^{-1}}$
c_3	$1.281 \cdot 10^{-1}$	€$\mathrm{J^{-1}}$
r	20.93^*	€$\mathrm{kg^{-1}}$
λ_2	$5 \cdot 10^{-5}$	€$(\mathrm{ppm\ m^2})^{-1}$
λ_3	$5 \cdot 10^{-3}$	€$(\mathrm{°C\,m^2})^{-1}$
λ_4	$7 \cdot 10^{-4}$	€$(\%\,\mathrm{m^2})^{-1}$

* Corresponding to 1.07 €$\mathrm{kg^{-1}}$ for fresh weight.

Table 3. Input and output constraints and initial conditions

Symbol	Value	Symbol	Value	Symbol	Value
$u_1^{\min}$	0	$u_1^{\max}$	1.2	$u_1(0)$	0
$u_2^{\min}$	0	$u_2^{\max}$	7.5	$u_2(0)$	0
$u_3^{\min}$	0	$u_3^{\max}$	150	$u_3(0)$	50
$x_1^{\min}$	0	$x_1^{\max}$	∞	$x_1(0)$	0.0035
$x_2^{\min}$	500	$x_2^{\max}$	1600	$x_2(0)$	0.001
$x_3^{\min}$	10	$x_3^{\max}$	20	$x_3(0)$	15
$x_4^{\min}$	0	$x_4^{\max}$	80% (78%)*	$x_4(0)$	0.008

* Constraint tightening of 2% for the stochastic case.

Deciding the harvest time is also an important factor in profitability. However, in this study, a fixed growing period of 40 days is chosen instead of including this decision in the optimization problem. This approach aligns with other studies in the literature (Boersma et al. (2022); Morcego et al. (2023); Mallick et al. (2025)). At the conclusion of this growing period, the crop is harvested and sold, marking the end of the cultivation cyle. The revenue generated from the sale is then used to calculate the Economic Profit Indicator (EPI), which is determined by subtracting the heating and CO_2 costs from the total earnings, as follows:

$$EPI = r\,x_1(t_f) - \sum_{k=0}^{t_f}\big(c_1u_1(k) + c_3u_3(k)\big)\Delta t, \tag{3}$$

where t_f denotes the final time, set at 40 days, c_1 represent the CO_2 injection cost coefficient, c_3 the heating cost coefficient, r the revenue coefficient from lettuce sales, and Δt the sample period. The values of these pricing factors are listed in Table 2.

The EPI defines the objective function for the greenhouse control problem. However, directly optimizing (3) using MPC or RL is difficult due to the sparse reward structure, since rewards are given only at the end of the growing period. To overcome this challenge, the following economic stage cost is defined for each time step:

$$\begin{aligned}\ell_e\big(x(k), u(k)\big) = &-r\big(x_1(k) - x_1(k-1)\big)\\ &+\big(c_1u_1(k) + c_3u_3(k)\big)\Delta t.\end{aligned} \tag{4}$$

The cumulative sum of the stage costs in (4) over the growing period is equivalent to the EPI in (3), but with the opposite sign. State constraints are essential to ensure the system operates within realistic and feasible bounds. Temperature, humidity, and CO_2 concentration must stay within specified minimum and maximum limits. However, due to inherent uncertainties, these constraints cannot be strictly enforced. Instead, they are treated as soft constraints by incorporating three penalty terms into the objective function to account for any violations. As a result, the following stage cost function is defined:

$$\ell\big(x(k),u(k)\big) = \ell_e\big(x(k),u(k)\big) + g_2\big(x_2(k)\big) + g_3\big(x_3(k)\big) + g_4\big(x_4(k)\big), \tag{5}$$

where g_2, g_3, g_4 are the penalty functions for the indoor CO_2 concentration, temperature, and humidity, respectively. These penalty functions are defined as

$$g_i\big(x_i(k)\big) = \begin{cases} \lambda_i\big(x_i(k) - x_i^{\max}\big) & \text{if } x_i(k) > x_i^{\max}, \\ \lambda_i\big(x_i^{\min} - x_i(k)\big) & \text{if } x_i(k) < x_i^{\min}, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The penalty coefficients λ_2, λ_3, and λ_4 are defined in Table 2. The state's minimum and maximum threshold values $x_i^{\min}$ and $x_i^{\max}$ are defined in Table 3. In addition to the state constraints, the control inputs are also subject to constraints. The heating injection, ventilation, and CO_2 injection are bounded by their respective minimum and maximum values, as defined in Table 3. Moreover, the control inputs are subject to rate constraints to prevent abrupt changes in the control actions. For each control input $i = \{1,2,3\}$, the absolute rate of change is limited to $\delta u_i^{\max} = u_i^{\max}/10$.

Considering the objective function in (5), along with the state and rate constraints, the optimal greenhouse control task is described by the following optimization problem:

$$\min_{u(\cdot)} \quad \sum_{k=0}^{t_f} \ell\big(x(k),u(k)\big), \tag{7a}$$

$$\text{s.t.} \quad x(k+1) = f\big(x(k),u(k),d(k),p\big), \tag{7b}$$

$$u_{\min} \le u_i(k) \le u_{\max}, \tag{7c}$$

$$|u(k) - u(k-1)| \le \delta u_{\max}. \tag{7d}$$

The RL agent, MPC and RL-Guided MPC all seek to solve the optimization problem in (7) and are ultimately evaluated based on the cumulative cost in (7a). The initial conditions, detailed in Table 3, were kept constant for every episode and for both the deterministic and stochastic cases.

3. REINFORCEMENT LEARNING

Actor-critic methods combine elements of both policy-based and value-based approaches, enabling stable and efficient learning. The actor is responsible for learning a policy, while the critic evaluates the quality of actions taken. By leveraging this dual structure, actor-critic algorithms allow policies to improve based on both direct experience and feedback from the critic. One key advantage of actor-critic algorithms is their ability to handle continuous state and action spaces effectively (Sutton and Barto (2020)), in contrast to other methods, such as Q-learning, that are better suited to discrete environments. This makes them well-suited for complex control tasks, such as greenhouse climate optimization. The actor-critic algorithm employed in this study is soft actor-critic (SAC) (Haarnoja et al. (2018)), a widely used actor-critic algorithm. SAC introduces an entropy term into the reward function, explicitly balancing exploration and exploitation.

Table 4. Hyperparameters for the RL agent

Parameter	Value
Training episodes	100
Warm-up episodes	9
Hidden layers	2
Neurons per hidden layer	128
Batch size	1024
Learning rate	$5 \cdot 10^{-3}$
Buffer size	100,000
Activation function	`ReLU`
Discount factor γ	0.95

This leads to the learning of a policy $\pi_\theta(s(k))$, function of the current observation $s(k)$ and parameterized by θ, where the entropy term determines policy randomness. By optimizing both the expected cumulative reward and entropy, the agent encourages diverse action selection.

Careful selection of the agent's observation space is crucial for effective learning. Providing insufficient information may hinder the agent's ability to learn meaningful strategies, while excessive information can make it difficult to extract relevant patterns. In a real-world greenhouse setting, expert growers do not have direct access to the current dry weight of the lettuce crop. However, in this study, it is assumed that this value, denoted by $x_1(k)$, is available to the RL agent, to MPC and to RL-Guided MPC. The agent also receives data on the greenhouse's indoor temperature $x_2(k)$, CO_2 concentration $x_3(k)$, and humidity $x_4(k)$, which are typically measured by sensors in real-world greenhouses. Moreover, the previous input $u(k-1)$ is included in the observation space. This information is crucial, since the difference between the current and previous control inputs is constrained, by (7d), to prevent abrupt changes. The weather disturbance $d(k)$ is also available to the agent, as it is for the MPC and for the RL-Guided MPC. Finally, the agent is designed to be time-aware, meaning that the current time step k is explicitly provided. This allows the agent to make decisions based on the stage of the growing period. Without this information, the agent would not be able to distinguish between early and late stages of the growing period, which is crucial for effective control. Therefore, the observation space is defined as:

$$s(k) = \big(x(k), u(k-1), d(k), k\big). \tag{8}$$

To ensure that the control input $u(k)$ adheres to the constraints specified in (7), the agent's action, denoted as $a(k) = \pi_\theta(s(k))$ with $a \in [-1,1]^3$, is interpreted as an adjustment to the previous control input:

$$u(k) = \max\big(u_{\min}, \min(u_{\max}, u(k-1) + a(k)\delta u_{\max})\big). \tag{9}$$

The reward function that the RL agent is trained to maximize is the same objective function defined in (5) but with an inverted sign. This formulation ensures that both methods operate under a consistent optimization framework, allowing for direct comparisons between all approaches. The discount factor, γ, is a crucial hyperparameter. For long-term tasks, $\gamma = 1$ allows the agent to consider the entire growth period and optimize long-term economic outcomes. However, it can destabilize training. To address this, we used $\gamma = 0.95$, which improved policy effectiveness. Table 4 lists all RL agent hyperparameters. Two different RL agents were trained, one for the deterministic environment and the other for the stochastic environment. Both agents were trained using identical hyperparameters, with the only difference being the environment, and consequently, the datasets they were exposed to during training.

4. MODEL PREDICTIVE CONTROL

The MPC controller in this study is designed to solve the optimization problem defined in (7). At time step k_0, the following optimization problem is solved:

$$\min_{u(k)} \quad \sum_{k=k_0}^{k_0+N_p} \ell\big(x(k), u(k)\big), \tag{10a}$$

$$\text{s.t.} \quad x(k+1) = f(x(k), u(k), d(k), p), \tag{10b}$$

$$u_{\min} \le u(k) \le u_{\max}, \tag{10c}$$

$$|u(k) - u(k-1)| \le \delta u_{\max}, \tag{10d}$$

$$x(k_0) = x_{k_0}, \tag{10e}$$

where N_p is the prediction horizon, and x_{k_0} is the system's state at time k_0. Once the optimization problem is solved, the first control input is applied to the system, and the optimization problem is solved again at the next time step. This process is repeated at each time step until the end of the growing period. Moreover, to help with the feasibility of the optimization problem, the solver is warm-started with the previous solution. The optimization problem is solved numerically using the open-source software framework CasADi (Andersson et al. (2019)) and the IPOPT solver (Wächter and Biegler (2006)).
Two different MPC controllers are designed, one for the deterministic environment and the other for the stochastic environment. The MPC formulation in (10) is used for both environments. The only modification for the stochastic case is a tighter indoor humidity constraint $x_4^{\max}$, 78% instead of 80%. This constraint tightening in the MPC formulation reduces the risk of violating the actual constraint at 80% in the stochastic simulation environment.

5. RL-GUIDED MPC

In the following, we outline the RL-Guided MPC framework, which incorporates the agent trained in Section 3 into the MPC optimization problem of Section 4. First, we outline how the terminal cost is derived from the critic. Next, we describe how the actor is utilized to define the terminal region constraint. Finally, we illustrate how the actor helps to provide a warm-start initial guess for the RL-Guided MPC optimization problem.

5.1 Terminal cost

In the RL-Guided MPC framework, the terminal cost is directly obtained from the actor network of the SAC agent trained in Section 3. Using the learned policy π_θ, multiple closed-loop trajectories are generated. This data is then used to train a cost function approximator $\tilde{J}_\phi$ that estimates the expected return for a given state. This approach enables the RL-Guided MPC to account for long-term effects over the entire growing period without extending the prediction horizon.
The cost function approximator is trained through expected return learning. Given a nominal state trajectory

Table 5. Hyperparameters for the cost function approximator

Parameter	Value
Hidden layers	2
Neurons per hidden layer	128
Batch size	1024
Learning rate	$1 \cdot 10^{-3}$
Buffer size	1024
Activation function	`tanh`

$x^{\mathrm{n}}(k)$, 1000 time steps are uniformly sampled from the interval $\{0, 1, \dots, t_f\}$. For each sampled time step $k^{(i)}$, an initial state $x(k^{(i)})$ is sampled from a uniform distribution within a range defined by the nominal trajectory:

$$x(k^{(i)}) \sim \mathcal{U}\big(x^{\mathrm{n}}_{\min}(k^{(i)}), x^{\mathrm{n}}_{\max}(k^{(i)})\big) \tag{11}$$

where

$$\begin{aligned} x^{\mathrm{n}}_{\min}(k) &= x^{\mathrm{n}}(k)(1-\sigma), \\ x^{\mathrm{n}}_{\max}(k) &= x^{\mathrm{n}}(k)(1+\sigma). \end{aligned} \tag{12}$$

The parameter σ, set at 50%, determines the spread of the sampled initial states around the nominal trajectory.
Next, from each sampled state $x(k^{(i)})$, the respective agent's observation $s(k^{(i)})$, defined in (8), is constructed and the policy π_θ is used to obtain a closed-loop trajectory that extends until the end of the growing cycle t_f. The cumulative cost of each trajectory is then defined as:

$$J_{\pi_\theta}(s(k^{(i)})) = \sum_{k=k_s}^{t_f} \ell\big(x(k), \pi_\theta(s(k))\big). \tag{13}$$

The cumulative cost of each trajectory is collected together with its respective dry weight $x_1(k^{(i)})$ and sampled time $k^{(i)}$ to form dataset $\mathcal{D}$. This dataset is divided into training and validation sets, with 80% of the data used for training and 20% for validation. To construct the cost function approximator $\tilde{J}_\phi$, a neural network parameterized by ϕ is trained using the values of $x_1(k^{(i)})$ and $k^{(i)}$ as inputs and the values of $J_{\pi_\theta}(s(k^{(i)}))$ as targets. The following loss function is minimized using the Adam optimizer (Kingma and Ba (2017)):

$$\mathcal{L}(\phi, \mathcal{D}) = \frac{1}{N_s} \sum_{i=1}^{N_s} \left(\tilde{J}_\phi(x_1(k^{(i)}), k^{(i)}) - J_{\pi_\theta}(s(k^{(i)})) \right)^2 .$$

The hyperparameters used for the cost function approximator are indicated in Table 5.
The cost function approximator is then added as a term to the objective function of the MPC optimization problem defined in (10). At each time step k_0, the cost function for the RL-Guided MPC is thus defined as

$$\sum_{k=k_0}^{k_0+N_p} \ell\big(x(k), u(k)\big) + \tilde{J}_\phi\big(x_1(k_0+N_p), k_0+N_p\big). \tag{14}$$

5.2 Terminal region constraint and initial guess

For the first time step, the terminal region constraint and the initial guess are constructed using the initial conditions, defined in Table 3, and the actor's policy. The agent's observation $s(0)$ is constructed using the initial conditions. Policy π_θ is then used to compute the closed-loop trajectory from the initial time step to $N_p + 1$:

$$\{s(0), s(1), \dots, s(N_p+1)\}, \tag{15}$$

and its respective control input trajectory:

$$\{u(0), u(1), \ldots, u(N_p)\}. \tag{16}$$

where $u(k) = \pi_\theta(s(k))$. From the computed closed-loop trajectory in the observation space defined in (15), the respective state trajectory is obtained:

$$\{x(0), x(1), \ldots, x(N_p + 1)\}. \tag{17}$$

The center of the terminal region constraint for the next optimization problem is then defined as the last state in this trajectory, i.e. $x_f = x(N_p + 1)$. The terminal region constraint for this initial optimization problem is then defined as

$$[(1 - \epsilon)x_f, (1 + \epsilon)x_f], \tag{18}$$

where ϵ, expressed as a percentage, defines the size of the terminal region around x_f. For the warm-start of the next optimization problem, the last N_p elements of the input trajectory defined in (16) are used as the initial guess. This terminal region constraint and initial guess construction is only used for the first time step. For subsequent time steps, the terminal region constraint and initial guess are constructed using the solution of the previous optimization problem and the actor's policy. The solution of the RL-Guided MPC optimization problem at each time step k_0 is denoted as

$$\{u(k_0), u(k_0 + 1), \ldots, u(k_0 + N_p - 1)\}. \tag{19}$$

and the corresponding predicted states are denoted as

$$\{x(k_0 + 1), x(k_0 + 2), \ldots, x(k_0 + N_p)\}. \tag{20}$$

The terminal region constraint and the initial guess for the next optimization problem are constructed using these solutions and the actor's policy. Starting from the final predicted state $x(k_0 + N_p)$, the agent's observation $s(k_0 + N_p)$ is constructed and policy π_θ is used to obtain the center of the terminal region for the next optimization problem:

$$x_f = f(x(k_0 + N_p), \pi_\theta(s(k_0 + N_p)), d(k_0 + N_p), p). \tag{21}$$

The terminal region constraint for the next optimization problem is then defined as in (18). The initial guess for the next optimization problem is constructed as

$$\begin{aligned}\big\{u(k_0 + 1), &u(k_0 + 2), \ldots, \\ &u(k_0 + N_p - 1), \pi_\theta\big(s(k_0 + N_p)\big)\big\}.\end{aligned} \tag{22}$$

Two RL-Guided MPC controllers were developed, one for the deterministic environment and the other for the stochastic environment. These controllers differ in the policy used to construct the terminal region constraint and the initial guess. Moreover, for the stochastic case, the indoor humidity constraint $x_4^{\max}$ is tightened to 78% to account for model uncertainties, reducing the risk of constraint violations, as for the MPC for the stochastic case in Section 4.

6. RESULTS

The performance of the RL agent, MPC, and RL-Guided MPC is evaluated in both the deterministic and stochastic environments. The results for an additional variant of RL-Guided MPC are also presented in the deterministic environment. For each environment, the three methods are compared based on the EPI, defined in (3), and the cumulative reward, equal to (7a) with the oppposite sign. For the figures in this section, we use the name 'RL-MPC' for the RL-Guided MPC for the sake of brevity.

6.1 Deterministic case

For the deterministic case, an additional variant of RL-Guided MPC is evaluated. This variant, denoted as 'RL-MPC-VFO', is identical to RL-MPC, except that the terminal region constraint is not included in the optimization problem. The simulation results for the deterministic environment are shown in Figure 1. In the EPI comparison, MPC and RL-Guided MPC exhibit similar performance across all prediction horizons, though RL-Guided MPC slightly outperforms MPC at shorter horizons. In contrast, the RL agent performs significantly worse than both methods. The RL-Guided MPC variant without the terminal region constraint performs similarly to MPC.

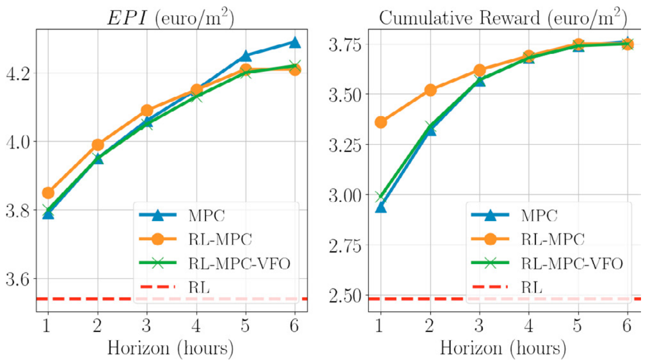

Fig. 1. Comparison of EPI and cumulative reward between RL, MPC, and RL-Guided MPC, at different prediction horizons, for the deterministic case.

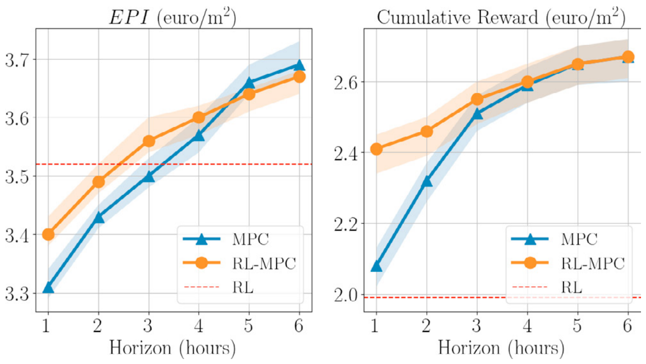

Fig. 2. Comparison of EPI and cumulative reward between RL, MPC, and RL-Guided MPC, at different prediction horizons, for the stochastic case.

Although comparing controllers using EPI is informative, all controllers aim to maximize the cumulative reward. In the cumulative reward comparison, a clear distinction emerges between MPC and RL-Guided MPC at lower prediction horizons, where RL-Guided MPC significantly outperforms MPC. This advantage stems from the RL-Guided MPC's ability to leverage the learned policy to account for long-term effects. However, as the prediction horizon increases, the difference in cumulative rewards diminishes, and beyond four hours, both methods exhibit similar performance. Meanwhile, the RL agent continues to perform significantly worse due to its difficulty in handling constraints effectively. In contrast, MPC and

RL-Guided MPC are explicitly designed to manage constraints, allowing them to operate near constraint boundaries without incurring penalties. The RL-Guided MPC variant without the terminal region constraint performs similarly to MPC. This result highlights the importance of the terminal region constraint in the RL-Guided MPC framework, which enables the agent to guide the optimization problem without relying solely on the learned cost function approximator.

6.2 Stochastic case

For the stochastic case, 30 different realizations of parameter $\hat{p}$, in (2), were generated. For each realization, the RL agent, MPC, and RL-Guided MPC were evaluated and the results were averaged. The outcomes are shown in Figure 2, where the shaded regions indicate the range between the minimum and maximum values. The results for the stochastic case show similar trends to the deterministic case, with some small differences. Considering the EPI comparison, RL-Guided MPC outperforms MPC at short prediction horizons. Unlike the deterministic case, the RL agent surpasses both MPC and RL-Guided MPC in terms of EPI at short horizons. Consistent with the deterministic case, the cumulative reward analysis confirms RL-Guided MPC's advantage over MPC at shorter horizons, particularly showing a notable improvement at a one-hour horizon.

7. CONCLUSIONS

This paper introduced the RL-Guided MPC framework for autonomous greenhouse control, combining the strengths of RL and MPC. By training an RL agent and integrating its learned policy into the MPC formulation through the construciton of a terminal cost and terminal region constraint, the proposed approach leverages RL's robustness to uncertainties and MPC's constraint-handling capabilities. Simulations in deterministic and stochastic environments demonstrated that RL-Guided MPC outperforms standalone RL and MPC with shorter prediction horizons. These results emphasize the value of embedding learned policies into MPC for enhanced performance, without the need to extend prediction horizons. However, it is important to note that the RL policy trained in this study did not perform as well as expected, with the cumulative reward being lower than that of the other methods, for every prediction horizon. In future work, we aim to investigate the potential impact of a better-performing policy, which we expect to result in substantial improvements. Additionally, using a policy derived from RL algorithms is not the only approach. There are various ways to obtain a suitable policy. For greenhouse control, for example, an expert knowledge rule-based controller could serve as an alternative policy, potentially offering valuable insights and improved performance.

ACKNOWLEDGEMENTS

The authors thank Sjoerd Boersma and Bart van Laatum for their valuable feedback on this work. The authors acknowledge the use of computational resources of the DelftBlue supercomputer, provided by Delft High Performance Computing Centre (https://www.tudelft.nl/dhpc).

REFERENCES

Amrit, R., Rawlings, J.B., and Angeli, D. (2011). Economic optimization using model predictive control with a terminal cost. *Annual Reviews in Control*, 35(2), 178–186. doi:10.1016/j.arcontrol.2011.10.011.

Andersson, J.A.E., Gillis, J., Horn, G., Rawlings, J.B., and Diehl, M. (2019). CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation*, 11(1), 1–36. doi:10.1007/s12532-018-0139-4.

Angeli, D., Amrit, R., and Rawlings, J.B. (2012). On average performance and stability of economic model predictive control. *IEEE Transactions on Automatic Control*, 57(7), 1615–1626. doi:10.1109/TAC.2011.2179349.

Boersma, S., Sun, C., and Van Mourik, S. (2022). Robust sample-based model predictive control of a greenhouse system with parametric uncertainty. *IFAC-PapersOnLine*, 55(32), 177–182. doi:10.1016/j.ifacol.2022.11.135.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Int. conference on machine learning (PMLR)*, 80:1861–1870.

Kingma, D.P. and Ba, J. (2017). Adam: A method for stochastic optimization. URL `https://arxiv.org/abs/1412.6980`.

Mallick, S., Airaldi, F., Dabiri, A., Sun, C., and De Schutter, B. (2025). Reinforcement learning-based model predictive control for greenhouse climate control. *Smart Agricultural Technology*, 10, 100751. doi:10.1016/j.atech.2024.100751.

Morcego, B., Yin, W., Boersma, S., Van Henten, E., Puig, V., and Sun, C. (2023). Reinforcement Learning versus Model Predictive Control on greenhouse climate control. *Computers and Electronics in Agriculture*, 215, 108372. doi:10.1016/j.compag.2023.108372.

Risbeck, M.J. and Rawlings, J.B. (2020). Economic Model Predictive Control for Time-Varying Cost and Peak Demand Charge Optimization. *IEEE Transactions on Automatic Control*, 65(7), 2957–2968. doi:10.1109/TAC.2019.2939633.

Sutton, R.S. and Barto, A. (2020). *Reinforcement learning: an introduction*. Cambridge: MIT Press.

Van Henten, E.J. (1994). *Greenhouse climate management: an optimal control approach*. Ph.D. thesis, Wageningen University.

van Laatum, B., van Henten, E.J., and Boersma, S. (2024). Greenlight-gym: A reinforcement learning benchmark environment for greenhouse crop production control. URL `https://arxiv.org/abs/2410.05336`.

Wächter, A. and Biegler, L.T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1), 25–57. doi:10.1007/s10107-004-0559-y.