agriculture

MDPI

Article

Energy-Saving Control Algorithm of Venlo Greenhouse Skylight and Wet Curtain Fan Based on Reinforcement Learning with Soft Action Mask

Lihan Chen, Lihong Xu * and Ruihua Wei

College of Electronics and Information Engineering, Tongji University, Cao'an Road, No. 4800, Shanghai 201804, China
* Correspondence: xulihong@tongji.edu.cn; Tel.: +86-136-363-62684

Abstract: Due to the complex coupling of greenhouse environments, a number of challenges have been encountered in the research of automatic control in Venlo greenhouses. Most algorithms are only concerned with accuracy, yet energy-saving control is of great importance for improving economic benefits. Reinforcement learning, as an unsupervised machine learning method with a framework similar to that of feedback control, is a powerful tool for autonomous decision making in complex environments. However, the loss of benefits and increased time cost in the exploration process make it difficult to apply it to practical scenarios. This work proposes an energy-saving control algorithm for Venlo greenhouse skylights and wet curtain fan based on Reinforcement Learning with Soft Action Mask (SAM), which establishes a trainable SAM network with artificial rules to achieve sub-optimal policy initiation, safe exploration, and efficient optimization. Experiments in a simulated Venlo greenhouse model show that the approach, which is a feasible solution encoding human knowledge to improve the reinforcement learning process, can start with a safe, sub-optimal level and effectively and efficiently achieve reductions in the energy consumption, providing a suitable environment for crops and preventing frequent operation of the facility during the control process.

Keywords: greenhouse control; energy saving; reinforcement learning; learning with knowledge

Citation: Chen, L.; Xu, L.; Wei, R. Energy-Saving Control Algorithm of Venlo Greenhouse Skylight and Wet Curtain Fan Based on Reinforcement Learning with Soft Action Mask. *Agriculture* **2023**, *13*, 141. https://doi.org/10.3390/agriculture13010141

Academic Editor: Panos Panagakis

Received: 6 November 2022
Revised: 8 December 2022
Accepted: 28 December 2022
Published: 5 January 2023

1. Introduction

The Venlo-type greenhouse originates from the Netherlands. It is a kind of small-roofed glass greenhouse that has become the most widely used type of glass greenhouse in the world. It has the characteristics of small cross-sectional members, convenient installation, high light transmission rate, good sealing, and a large ventilation area. A typical Venlo-type greenhouse is shown in Figure 1. The equipment of a Venlo-type greenhouse generally includes skylights, wet curtain fans, sunshades, heating equipment, etc. Owing to the interaction of greenhouse microclimate factors, including light, temperature, humidity, carbon dioxide concentration, etc., the design of the control algorithm for a Venlo-type greenhouse faces some challenges. In practical application, the multi-factor coordinated control algorithm [1–3] effectively decouples control from the action of the actuator, but relies on experience and repeated experiments to set control parameters, and the parameters of the algorithm need to be adjusted manually, often with seasonal changes. Additionally, the energy consumption of greenhouse control facilities is seldom considered in the control algorithm. In the context of a greenhouse, higher control accuracy is bound up with greater energy consumption. Considering the comprehensive benefits of greenhouse environmental control, it is extremely important to reduce energy consumption on the premise of a comparatively suitable microclimate for crops. The energy consumption of heating equipment in the cold season and the electrical energy consumption of wet curtain fans for cooling in hot climates are the main components of consumption in this type of

greenhouse, followed by the electrical energy consumption of the motor of the skylight and other facilities in the switching stroke.

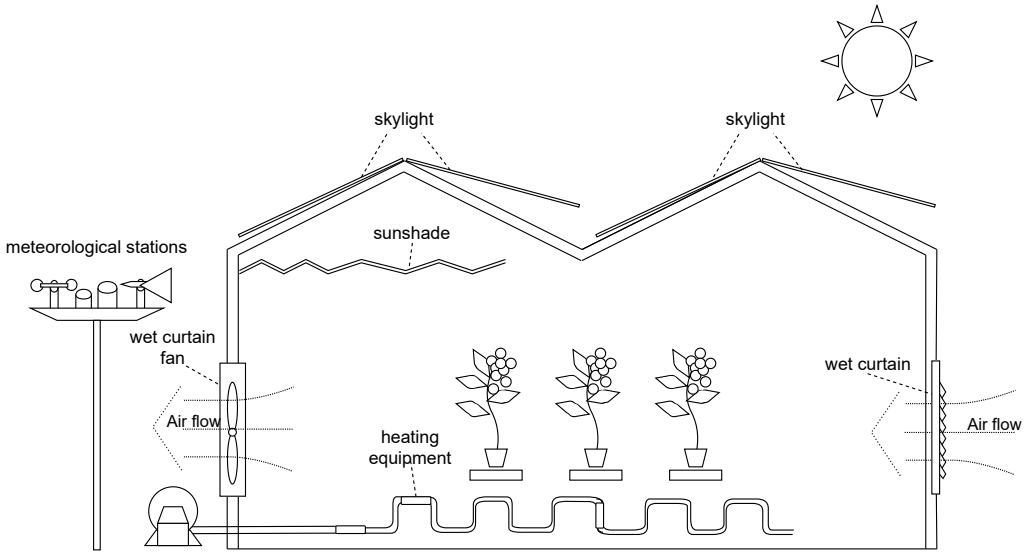


Figure 1. The overview of a typical Venlo-type greenhouse with main facilities.

This paper conducted in-depth research on the energy-saving optimization problem of the skylight and the wet curtain fan for cooling control in a hot climate. The skylight cools the indoor air through indoor and outdoor air heat exchange, while the wet curtain fan system forms a low-temperature airflow that speedily passes through the wet curtain wall and carries moisture from the curtain into the room through fan exhaust to achieve cooling; its cooling capacity is stronger than the skylight's. Nonetheless, when the wet curtain fan is opened, the skylight needs to be closed to ensure the effect, and the actions between the two are mutually exclusive. As a result, our optimization goal is to ensure that the factors in the greenhouse are comparatively appropriate, to shorten the use time of the fan, and to avoid the frequent action of the sunroof resulting from the mutual exclusion of the actions of the two mechanisms.

Some researchers have applied multi-objective optimization [4–6] or model prediction methods [7,8] to optimize energy conservation, but the proposed algorithms usually have problems such as the high difficulty of solving and online optimization, as well as an over-reliance on model prediction accuracy [9]. In recent years, intelligent methods have gradually been adopted in the field of energy consumption model prediction and energy conservation control [10–13], where reinforcement learning is a new idea to explore optimization policies in complex environments. It continuously optimizes the multi-objective strategy through the rewards and punishments obtained by interacting with the unknown environment. In the field of greenhouse control, several researchers [14–16] optimized heating, fruit yield, and planting cost while maintaining effective control over greenhouse factors. Andrey et al. [17] and Afzali et al. [18] reduced the average light supplement duration on the basis of maintaining the performance of the leaf area. Nonetheless, most of the above research is on the basis of offline simulation experiments that are dissimilar from the actual complex environment. The direct use of the control strategies obtained in the actual environment often has poor benefits, but the reinforcement learning process that directly interacts with the actual environment online requires a lot of time and benefit costs for early training, and even unsafe actions may occur, causing damage to the facilities.

For the practical application of reinforcement learning, relevant theoretical research has also developed a multitude of branches. For instance, offline reinforcement learning aims to conduct pre-training through offline data obtained from existing policy interactions [19–24], and security reinforcement learning aims to ensure the security of the final policy and even the exploration process by defining the cost function [25], focusing on bad samples [26], or

controlling stability theory [27]. Nevertheless, the offline learning effect is limited by the sample quality, and most of the safe learning does not guarantee the safety of the training process. Methods to explore the safety of the process [27] are mostly theoretical on account of the strong assumptions based on the object model.

As a model-less method, although its network structure has a strong learning ability for complex problems, the initialization and exploration of policies are random, which is the main reason for the low and unstable early benefits of this method. However, in practical terms, part of the knowledge about the environment is summarized from the past, which can be used to guide the policy initialization and even optimization process, avoiding low returns and dangerous actions from the initial stage of learning, reducing the time cost, and speedily improving the benefits. At present, it is feasible to reconstruct the policy into a specific space constrained by rules [28,29] or initialize the policy network based on a decision tree [30]. Nonetheless, the former rule space is determined before learning, while the second method's learning effect depends on the design quality of the initial decision tree, which requires perfect rules and is difficult to apply to complex greenhouse environments.

Under the background of the above problems, this paper deals with the energy-saving control of a skylight and wet curtain fan over temperature and humidity in a Venlo-type greenhouse in summer by integrating some existing knowledge and rules into the reinforcement learning method to reduce energy consumption while ensuring environmental suitability. The control problem and the proposed approach are described in detail in the following Section 2. Section 2.1 starts with an introduction to the controlled microclimate factors in Venlo-type greenhouses, describing the summer control problem in Venlo-type greenhouses as well as the control objectives of the problem in a mathematical form. Section 2.2 presents the recommended approach, which is based on reinforcement learning combined with the structure of the Soft Action Mask (SAM) designed to enable knowledge-initiated reinforcement learning, a safe and efficient training process for reinforcement learning, and optimizable rules. Section 3 compares and analyzes the training results with several existing methods in terms of efficiency gains, control effectiveness, and safety. Section 4 summarizes the conclusions and implications of the work.

2. Method

2.1. Problem Description

Generally, in Venlo-type greenhouses, the mathematics of the change process can be described by Equation (1) [31], considering the three microclimate factors of the temperature, the humidity, and the concentration of CO_2 in the greenhouse, where $\boldsymbol{f}$ and $\boldsymbol{G}$ are unknown functional relations characterizing the mechanism of the control object, respectively:

$$\boldsymbol{F}(\boldsymbol{x}) = \frac{d\boldsymbol{x}}{dt} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{v}) + \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{v})\boldsymbol{u} \tag{1}$$

$$\begin{aligned} \boldsymbol{x} &= [T_{air}, H_{air}, CO_{2,air}]^T \\ \boldsymbol{v} &= \left[I_{glob}, T_{\text{out}}, H_{\text{out}}, CO_{2,out}, V_{\text{wind}}\right]^T \\ \boldsymbol{u} &= \left[u_{\text{vent}}, u_{\text{pad}}, u_{\text{scr}}\right]^T \end{aligned} \tag{2}$$

Variables in (2) are shown in Table 1 below.

The mechanism in a practical environment provides qualitative knowledge, but suffers from difficulties in decoupling and parameter determination. The skylight heat exchange $Q_{\text{vent}}\left[\frac{W}{m^2}\right]$ described in (3) is taken as an example. The explanation of symbols is shown in Table 1.

$$Q_{\text{vent}} = \rho_{\text{air}}\, C_{p,\,\text{air}}\, \varphi_{\text{vent}}\, (T_{\text{air}} - T_{\text{out}}) \tag{3}$$

Table 1. Explanation of symbols of variables and parameters.

Symbol	Explanation	Unit
$\boldsymbol{x}$	The indoor data	
T_{air}	The temperature of indoor air	°C
H_{air}	The absolute humidity of indoor air	$\frac{\text{g}}{\text{m}^3}$
$CO_{2,air}$	Indoor carbon dioxide concentration	$\frac{\text{mg}}{\text{m}^3}$
$\boldsymbol{v}$	The outdoor data	
T_{out}	The temperature of outdoor air	°C
H_{out}	The absolute humidity of outdoor air	$\frac{\text{g}}{\text{m}^3}$
I_{glob}	The light intensity	$\frac{\text{W}}{\text{m}^2}$
$CO_{2,\text{ out}}$	Outdoor carbon dioxide concentration	$\frac{\text{mg}}{\text{m}^3}$
V_{wind}	Outdoor wind speed	$\frac{\text{m}}{\text{s}}$
$\boldsymbol{u}$	The controller input vector	
u_{vent}	Skylight opening output	
u_{pad}	The wind speed scale of the wet curtain fan	
u_{scr}	Opening output of the shading screen	
Q_{vent}	The heat exchange by the skylight	$\frac{\text{W}}{\text{m}^2}$
ρ_{air}	The air density	$\frac{\text{kg}}{\text{m}^3}$
$C_{p,\text{ air}}$	The heat capacity	$\frac{\text{J}}{\text{K}\cdot\text{kg}}$
φ_{vent}	The ventilation flow rate	$\frac{\text{m}^3}{\text{m}^2\cdot\text{s}}$
$\boldsymbol{s}$	State of observation	
$\boldsymbol{a}$	Action space	
ΔT_{air}	The temperature change of indoor air	°C
ΔT_{out}	the temperature change of outdoor air	°C
t_{vent}	Time steps the skylight keeps one state	
t_{pad}	Time steps the fan keeps one state	

It is clear from this mechanism that a skylight can be opened to cool the room when the outdoor temperature is suitable, but it is difficult to quantify precisely the range of 'suitable' by summarizing this knowledge as a quantitative rule. In addition, the wet curtain fan forcibly draws out the indoor air by starting the fan to cause negative pressure, so that the curtain wall is wetted by cold water. This allows the airflow to pass quickly and carries moisture into the room in order to evaporate heat absorption to achieve cooling [32]. Generally, the cooling effect is related to the ventilation rate of the wet curtain fan, outdoor temperature and humidity, and the humidity difference between the wet curtain and the outdoor humidity. Therefore, high outdoor air humidity can cause poor cooling effects using wet curtain fans. The interaction of fans and skylights on the greenhouse environment easily gives rise to unnecessary frequent actions and energy consumption waste, making it difficult to formulate control parameters under multiple actuators and controlled factors. Considering that the main energy consumption of the wet curtain fan is generated by the fan that operates for a long time, while the skylight only generates a small amount of energy consumption in the opening and closing stroke, our goal is to reduce the fan's operation time and the change in sunroof control quantity as much as possible within the reasonable range of temperature and humidity control errors. The specific optimization goal is as follows (4):

$$J = \int_0^T u_{\text{pad}}(t)dt + \sum_{i=0}^{T} \Delta u_{\text{vent}} \tag{4}$$

The form of the problem to be optimized is shown in (5), where c_j refers to the microclimate suitability and the actuator safety constraints.

$$\begin{aligned} &\min_{u} J \\ &\text{s.t. } \boldsymbol{F}(\boldsymbol{x}) - \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{v}) - \boldsymbol{G}(\boldsymbol{x}, \boldsymbol{v})\boldsymbol{u} = 0 \\ &\boldsymbol{c}_j(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{v}) \leq 0, \mathrm{j} = 1, 2, \ldots, \mathrm{n} \end{aligned} \tag{5}$$

2.2. Algorithm

In a practical application of reinforcement learning, if an invalid action can be predicted in the current state, the Action Mask can be used to prune it to accelerate learning [33–35]. However, this method sets the probability of some actions directly to zero under particular situations, so too many hard rules result in too many limitations on exploration. It is difficult to design a set of perfect rules and parameters for a greenhouse control problem with complexly coupled factors, but empirically, some fuzzy conditions from experience can be summarized to determine whether a particular facility is currently suitable for use. Our method proposes the design of a network of masking rules that can be manually initialized by partially empirical rules and that can be updated with parameters based on feedback by a policy gradient algorithm during reinforcement learning, which was referred to as SAM.

2.2.1. Overview

The overall algorithmic architecture of the proposed method is shown in Figure 2. The agent includes the Actor and the Critic, and the Actor includes two networks: one fully connected feedforward neural network, which is originally randomly initialized, and one soft-action-mask network (in Section 2.2.2), which is manually initialized according to the rules. The output of the SAM network, which is a bias layer, biases the probability of the logits layer of the original neural network before the softmax activation function, suppressing the sampling of low-benefit actions and obtaining the final policy $\pi_\theta(a \mid s)$, which is the probability of each action and where θ is the parameter of completed Actor-network.

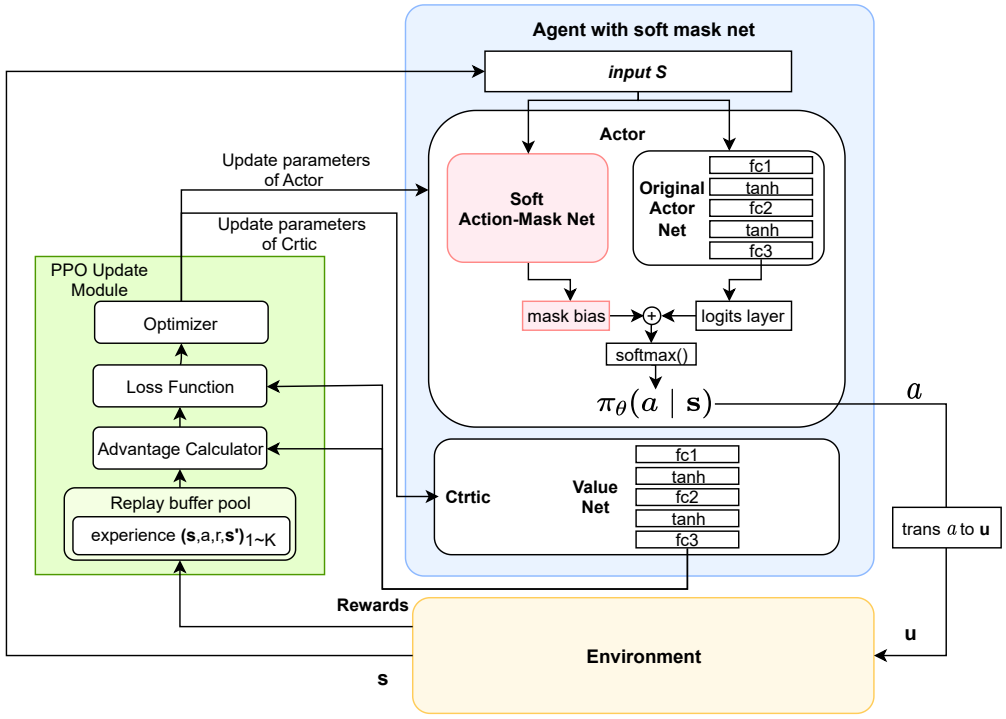


Figure 2. The overall architecture of our provided reinforcement learning method with SAM.

The form of each variable in Figure 2 is shown in (6), where $\boldsymbol{x}$, $\boldsymbol{v}$, and $\boldsymbol{u}$ are the same as in Equation (2). ΔT_{air} (°C) and ΔT_{out} (°C), observing the trend and the intensity of indoor and outdoor temperature change, denote the difference between the sampled temperature at the moment $\boldsymbol{t}$ and its previous moment. t_{vent} , t_{pad} are the time steps in which one state of the facilities lasts. $\boldsymbol{a}$ is the output of the Actor with dimension o_s, i.e., the set of actions to be selected by RL agent policy, and the final selected action a_i is sampled from the policy distribution.

$$\begin{cases} \boldsymbol{s} = \left[\boldsymbol{x}^T, \boldsymbol{v}^T, \boldsymbol{u}^T, \Delta T_{\text{air}}, \Delta T_{\text{out}}, t_{\text{vent}}, t_{\text{pad}}\right]^{\boldsymbol{T}} \\ \boldsymbol{u} = \left[u_{\text{pad}}, u_{\text{vent}}\right]^{\boldsymbol{T}} \\ \boldsymbol{a} = \{a_i\}_{i=0,1,\ldots,o_s-1} \end{cases} \tag{6}$$

$$\text{where} \begin{cases} \Delta T_{\text{out}} = T_{\text{out},t} - T_{\text{out},t-1} \\ \Delta T_{\text{air}} = T_{\text{air},t} - T_{\text{air},t-1} \end{cases}$$

After the action decoupling module, $\boldsymbol{a}$ is converted into the facilities' control input quantity $\boldsymbol{u}$. The agent continuously selects the action according to the current policy Actor and observes the state of the environment to collect the reward information from the historical feedback, which is put into the experience recovery pool of the PPO Update Loss module in Figure 2. After several rounds, based on the PPO algorithm [36], the agent's parameters are updated.

The PPO algorithm is based on the Actor–Critic framework [37] and invokes the concept of importance sampling to enhance efficiency. The loss function for policy optimization based on the advantage function is as shown in (7), where $\widehat{A}(s,a)$ is the estimated advantage. Forcing a constraint on the ratio of old to new probabilities in the update, the new loss function is described as in (8) and (9), where $r(\theta)$ is the ratio of the probability distribution of the updated policy π_θ to the old policy $\pi_{\theta_{\text{old}}}$, ε is a small positive constant close to zero, and the final loss function is as in (10), considering the error and entropy terms:

$$L^{PG}(\theta) = E_\pi\left[\ln \pi_\theta(a \mid s)\widehat{A}(s,a)\right] \tag{7}$$

$$L^{CLIP}(\theta) = E_\pi\left[\min\left(r(\theta)A_{\theta_{\text{old}}}(s,a), \text{clip}(r(\theta), 1-\varepsilon, 1+\varepsilon)\right)\widehat{A}_{\theta_{\text{old}}}(s,a)\right] \tag{8}$$

$$r(\theta) = \frac{\pi_\theta(a \mid s)}{\pi_{\theta_{\text{old}}}(a \mid s)} \tag{9}$$

$$L^{CLIP+VF+S}(\theta) = E_\pi\left[L^{CLIP}(\theta) - c_1\left(V_\theta(S) - V_{\text{target}}\right)^2 - c_2 H(s,\pi_\theta)\right] \tag{10}$$

When updating the parameters, the Actor estimates the advantage function by interactively collecting sample data over K time steps in (11), where λ is the discount factor of advantage, γ is the reward discount, δ_t is the TD error, r_t is the immediate reward at t, and V is the state value calculated based on the Critic network. V_{target} is the actual long-term cumulative benefit, usually estimated as (12):

$$\begin{cases} \widehat{A}_t(s,a) = \delta_t + (\gamma\lambda)\delta_{t+1} + \ldots + (\gamma\lambda)^{K-t+1}\delta_{K-1} \\ \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t) \end{cases} \tag{11}$$

$$V_{t\,\text{arget}} = \widehat{A}_t(s,a) + V(s_t) \tag{12}$$

The PPO algorithm flow pseudo-code is shown in Algorithm 1. The symbols and parameters involved are explained in Table 2.

Algorithm 1 PPO Algorithm

Input: total numbers of iterations N, max steps per round K,replay buffer pool of history experience RB,policy before update $\pi_{\theta_{old}}$, update epoch Z

for $episode = 1, 2, \ldots, N$ **do**
  for $step = 1, 2, \ldots, K$ **do**
    Get observation state s from environment
    Choose action a by policy $\pi_{\theta_{old}}$
    Run a in environment
    Observe reward r and new state $s^{'}$
    Add experience information $\{s,a,r,s^{'}\}$ into RB
  end for
  Calculate advantage $\hat{A}_1, \hat{A}_2, \ldots, \hat{A}_K$, with RB based on (11)
  Optimize loss function (8–10) and (12) with respect to θ for Z epochs
  Update θ_{old} to θ
end for

Table 2. Explanation of symbols in formulas.

Symbol	Explanation
π_θ	The updated policy
$\pi_{\theta_{\text{old}}}$	The old policy
ε	A small positive constant close to zero
N	Total numbers of iterations
K	Update every K time steps
Z	Optimize the parameters for Z epochs every K steps
λ	The discount factor of advantage
γ	The reward discount
δ_t	The TD error
r_t	The immediate reward at t
V	The state value calculated based on the Critic network
V_{target}	The actual long-term cumulative benefit
σ_i	The output of the judgment nodes
w_i, c_i	The initialization weights and bias parameters of the judgment nodes, respectively
α_i	The degree of uncertainty of the discriminant conditions
$\boldsymbol{P_{n\times k}}$	The path matrix initialized by $\boldsymbol{k}$ discriminative paths composed of $\boldsymbol{n}$ different conditions in series according to $\boldsymbol{k}$ rules
$\boldsymbol{M_{out}}$	The final output of the bias layer after SAM
I_s	The dimension of observation input space
O_s	The dimension of action output space

2.2.2. Soft Action-Mask Net

In the study [34], it was mentioned that the use of action masks allows the pruning of invalid actions. It was demonstrated that this process can be regarded as a state-dependent differentiable function [35], which is in line with the assumptions of the Policy Gradient Algorithm. In the greenhouse, there are also several invalid actions that can be predicted. Significantly, exploration in a real greenhouse requires avoiding the execution of dangerous actions in particular states in order to ensure the safety of the crops and the facilities.

To distinguish it from the SAM proposed later, this type of action mask for dangerous actions in greenhouses is referred to as a Hard Action Mask (HAM). Take action $\boldsymbol{a} = \{a_0, a_1, a_2, a_3\}$ for an example: if a_3 corresponds to the action of turning on the fan, it is not advisable to choose this action in low temperatures according to human experience. Assuming that state s_0 is in a low temperature, the logits output of the Actor is

$l = [l_0, l_1, l_2, l_3] = [1.0, 1.0, 1.0, 1.0]$ and the policy network output transformed into action sampling probabilities after softmax activation is $\pi_\theta(\boldsymbol{a} \mid \boldsymbol{s_0})$:

$$\begin{aligned}\pi_\theta(\boldsymbol{a} \mid \boldsymbol{s_0}) &= [\pi_\theta(a_0 \mid \boldsymbol{s_0}), \pi_\theta(a_1 \mid \boldsymbol{s_0}), \pi_\theta(a_2 \mid \boldsymbol{s_0}), \pi_\theta(a_3 \mid \boldsymbol{s_0})] \\ &= \text{softmax}([l_0, l_1, l_2, l_3]) \\ &= [0.25, 0.25, 0.25, 0.25]\end{aligned}$$

$$\text{where } \pi_\theta(a_i \mid \boldsymbol{s_0}) = \frac{e^{l_i}}{\sum_{i=0}^{o_s} e^{l_i}} \tag{13}$$

The HAM sets the logits corresponding to action a_3 to an extremely small value, Min, and the probability of selecting a_3 will be set to zero as:

$$\pi'_\theta(\boldsymbol{a} \mid \boldsymbol{s_0}) = \text{softmax}([l_0, l_1, \text{Min}, l_3]) = [0.33, 0.33, 0, 0.33] \tag{14}$$

HAM ensures the safe process by completely setting the sampling probability of unsafe or invalid actions to zero. However, exploration in the safe region does not guarantee good gains for all action trajectories. Furthermore, in addition to dangerous actions, low benefits in the early stages of learning are also undesirable, and initialization of a high level can effectively shorten the learning period. Actually, in realistic scenarios, instructive human knowledge can be provided beforehand. To combine knowledge with reinforcement learning, ref. [30] proposed a method of initializing neural networks in the form of decision trees to implement a warm start. However, the human experience summarized in greenhouses is usually ambiguous, and it is hard to cover the complete state domain. In addition, the multi-factor coupling relationship results in a complex of conditional rules, leading to difficulty in applying the results of [30], where the initialization of the decision tree is too complicated and the over-regulated exploration restricts boosting capabilities of the agent. We propose a method transforming several ruled soft masks into a network form and incorporating partial experience to guide the agent's exploration process, finally achieving reinforcement learning of better initialization for greenhouse control. The networked form is shown in Figure 3.

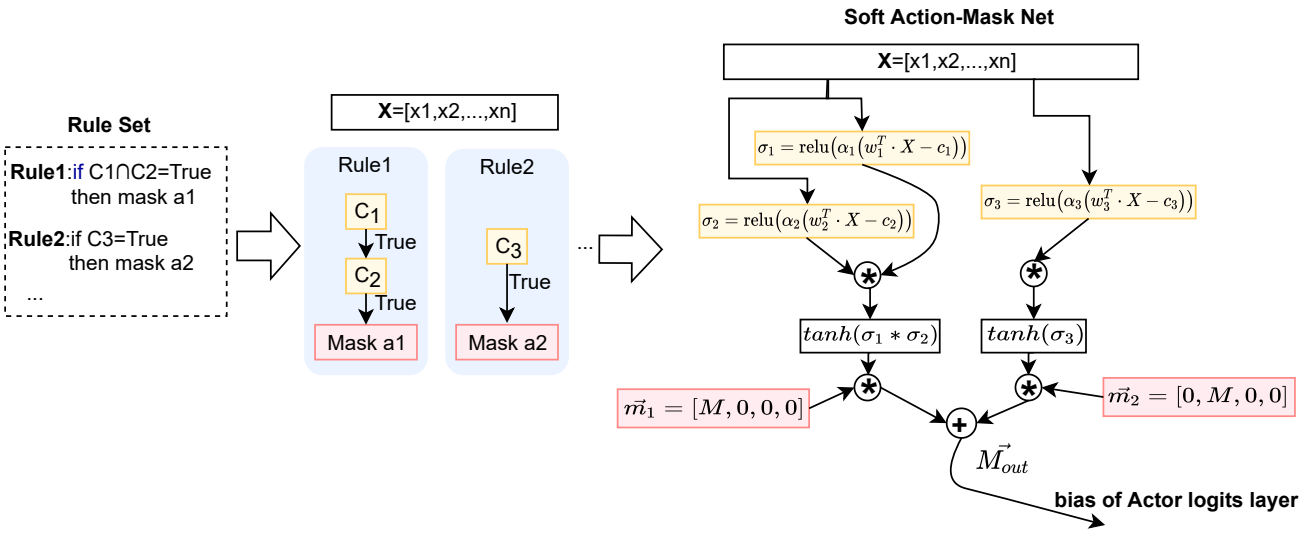


Figure 3. Initialization of Soft Action-Mask Net from the original rules set by human knowledge.

By manually initializing the network weights and comparators, the judicial process with combined first-order rules is transformed through the network into a bias layer that reduces the probability of sampling improper actions, which is referred to as SAM. Unlike

the direct masking of HAM, SAM adds a bias layer to the output of the logits layer of the original policy network of the agent. The offset base value is initialized to a relatively small value, avoiding full suppression of the policy gradient backpropagation found in HAM. Secondly, the rules set by manual experience are sometimes vague and one-sided. The networked SAM allows the weights and offsets of the rule nodes in the network to be dynamically trained through the gradient, allowing the intelligent agent to explore better policy and rule parameters through the reward mechanism.

The network initialization requires manual pre-establishment of the masking rules, and the rules build the network in the form of weights and comparators. The initialization process is Algorithm 2. Assuming a total of k rules consisting of n first-order discriminant conditions, as in Figure 3, σ_i is the output of the judgment nodes, w_i and c_i are the initialization weights and bias parameters of the judgment nodes, respectively, and α_i is the degree of uncertainty of the discriminant conditions, taken from 0 to 1. Equations (15)–(21) comprise the calculation process from the input to the bias layer, where $\boldsymbol{P_{n\times k}}$ is the path matrix initialized by k discriminative paths composed of n different conditions in series according to k rules, and $\boldsymbol{M_{out}}$ is the final output of the bias layer after SAM.

Algorithm 2 Initialize of SAM net

Input: Expert knowledge rules collection $R_{ini} = \{r_j\}$, numbers of rules k, count of all conditions n

initialize collection of path $\boldsymbol{P} = \{\}$
initialize collection of weight $\boldsymbol{W} = \{\}$
initialize collection of comparison $\boldsymbol{c} = \{\}$
initialize collection of mask $\boldsymbol{M} = \{\}$
for $j = 1, 2, \ldots, k$ **do**
    $\boldsymbol{p_j} = \{\}$
    $\boldsymbol{m_j} = \{\}$
    for $i = 1, 2, \ldots, n$ **do**
        if condition $C_i \in r_j$ **then**
            $\boldsymbol{W} = \boldsymbol{W} \cup \boldsymbol{w_i}$
            $\boldsymbol{c} = \boldsymbol{c} \cup c_i$
            initialize p_{ij} by (17)
            $\boldsymbol{p_j} = \boldsymbol{p_j} \cup p_{ij}$
        end if
    end for
    $\boldsymbol{P} = \boldsymbol{P} \cup \boldsymbol{p_j}$
    initialize $\boldsymbol{m_j}$ by (20-21)
    $\boldsymbol{M} = \boldsymbol{M} \cup \boldsymbol{m_j}$
end for

$$\boldsymbol{\sigma} = \{\sigma_i\}_{i=1,..,n} = \left\{\mathbf{relu}\left[\alpha_i\left(w_i^{\mathbf{T}}\boldsymbol{x} - c_i\right)\right]\right\}_{i=1,..,n} = \mathbf{relu}\left[\boldsymbol{\alpha}^{\mathbf{T}}\left(\boldsymbol{W}^{\mathbf{T}}\boldsymbol{x} - \boldsymbol{c}\right)\right] \tag{15}$$

$$\boldsymbol{B_{n\times k}} = \left\{b_{ij}\right\}_{\boldsymbol{n\times k}} = \mathrm{Filter}\left(\left[\sigma_i\right]_{i=1,\ldots,n} \cdot \boldsymbol{P_{n\times k}}\right) \tag{16}$$

$$\text{where} \begin{cases} \boldsymbol{P_{n\times k}} = \left\{p_{ij}\right\} = \begin{cases} 1, & \text{condition i in Rule } j \\ 0, & \text{otherwise} \end{cases} \\ \mathrm{Filter}\left(\mathbf{X}\right) = \begin{cases} 1, & x_{ij} = 0 \\ x_{ij}, & \text{otherwise} \end{cases} \end{cases} \tag{17}$$

$$\boldsymbol{BM_{1\times k}} = \left[\boldsymbol{BM_j}\right]_{\mathbf{1}\times\boldsymbol{k}} = \left[\mathbf{tanh}\left(\prod_i^n b_{ij}\right)\right] \tag{18}$$

$$\boldsymbol{M_{out}} = \left[\boldsymbol{m_1}, \ldots, \boldsymbol{m_j}, \ldots, \boldsymbol{m_k}\right] \cdot \boldsymbol{BM_{1\times k}^{\mathbf{T}}} \tag{19}$$

$$\boldsymbol{m_j} = \left[m_{o_d}\right]_{1\times o_s} = [m_0, m_1, \ldots, m_{o_s}] \tag{20}$$

$$\text{where } m_{o_d} = \begin{cases} M, \text{ if mask action o\_d in Rule } j \\ 0, \text{ otherwise} \end{cases} \tag{21}$$

Still considering the fan action, suppose there are rules as follows: when the indoor temperature is below 30 degrees Celsius and the outdoor temperature is below 28, try not to turn on the fan. State s $= [T_i, T_o, H_i, H_o]$, action space $a = \{a_0, a_1, a_2, a_3\}$, where a_3 corresponds to turning on the fan. Initialize $C_1 : T_i < 30 \rightarrow \begin{cases} w_1 = [-1, 0, 0, 0] \\ c_1 = -30 \end{cases}$, $C_2 : T_o < 28 \rightarrow \begin{cases} w_2 = [0, -1, 0, 0] \\ c_2 = -28 \end{cases}$, and $\boldsymbol{m_1} : [0, 0, 0, \mathrm{M}]$. When the condition is not met, the offset is 0, otherwise the larger the absolute value of the offset, the higher the degree of inhibition of the action, with less inhibition near the critical value of the judgment condition.

2.2.3. Region Reward

The reward function consists of two main components: a control interval error penalty R_{err} and a fan loss penalty R_{fan_on}, as shown in Equations (22) and (23), respectively. $c_{\text{err}}, c_{fan\_on}, c_{freq}$ are the weights for each, and $\boldsymbol{x}_{set}$ indicates the target, in which x_{set} is the target for each factor:

$$R_{err} = c_{err}|\boldsymbol{x} - \boldsymbol{x}_{set}|^2 \tag{22}$$

The fan loss penalty R_{fan_on} is an indirect measure of fan energy consumption based on the cumulative hours of the fan running, T is the data sampling period, and the element value x_{set} is determined based on the current state and the upper and lower limits $x_{\text{high\_com}}$ and $x_{\text{low\_com}}$ of suitability, as in (24):

$$R_{\text{fan\_on}} = c_{\text{fan\_on}} \int_0^T u_{\text{fan},t} dt \tag{23}$$

$$x_{set} = \begin{cases} x_{\text{high\_com}}, & x > x_{\text{high\_com}} \\ x_{\text{low\_com}}, & x < x_{\text{low\_com}} \end{cases} \tag{24}$$

Depending on the crop's growth requirements and tolerance, three levels of condition areas are distinguished. $\Re_{\text{comf}}$ is an excellent environmental condition for growth, $\Re_{\text{toleran}}$ is a sub-optimal condition that can be tolerated for a long period, and $\Re_{\text{extreme}}$ is an extreme condition that can only be tolerated for a short period. If the environment has difficulty in running at $\Re_{\text{comf}}$ with the lack of precise control equipment, then running in $\Re_{\text{toleran}}$ is expected and $\Re_{\text{extreme}}$ should be avoided at all costs. The setting of the penalty term is therefore differentiated by the current state, as shown in Equation (25):

$$R = \begin{cases} R_{\text{fan\_on}} + R_{\text{freq}} & , \quad x \subset \Re_{\text{comf}} \\ R_{\text{err}} + R_{\text{fan\_on}} + R_{\text{freq}} & , \quad x \subset \Re_{\text{toleran}} \\ R_{\text{err}} + p & , \quad x \subset \Re_{\text{extreme}} \end{cases} \tag{25}$$

The complete RL with SAM algorithm flow is shown in Algorithm 3.

Algorithm 3 RL with Soft Action-Mask

Input: Dimension of state observation I_s , dimension of action O_s, expert knowledge R_{ini}, total numbers of iterations N, max steps per round K, replay buffer pool of history experience RB, update epoch Z

randomly initialize actor and critic parameters in Agent
initialize mask net in Agent with R_{ini} by Algorithm 2
for $episode = 1, 2, \ldots, N$ **do**
  for $step = 1, 2, \ldots, K$ **do**
    get observation state $\boldsymbol{s}$ from environment
    get logits layer $\boldsymbol{l}$ forward the actor net
    get bias $\boldsymbol{M_{out}}$ forward the soft mask net by (15)–(21)
    $\pi_{\theta_{old}}(\boldsymbol{a}|\boldsymbol{s}) = softmax(\boldsymbol{l} + \boldsymbol{M_{out}})$
    choose action a by policy $\pi_{\theta_{old}}(\boldsymbol{a}|\boldsymbol{s})$
    run a in environment
    observe new state $\boldsymbol{s}'$ and calculate reward r by (22)–(25)
    add experience information $\{\boldsymbol{s}, a, r, \boldsymbol{s}'\}$ into RB
  end for
  update θ_{old} to θ for Z epochs by Algorithm 1
end for

3. Experimental Results and Discussion

A series of comparative experiments were conducted in a three-state simulation greenhouse model. The hyperparameters for the reinforcement learning process were selected as follows: $I_s = 11, O_s = 3, Z = 10, N = 1600, K = 288$, and learning rate = 6×10^{-4}.

3.1. RL Efficiency

Firstly, in terms of reward enhancement, the comparison methods included RL with a reward function alone, RL with HAM, and our proposed RL with SAM. Five learning sessions were performed for each learning method, and the results of recording their mean reward profile and standard deviation are shown in Figure 4.

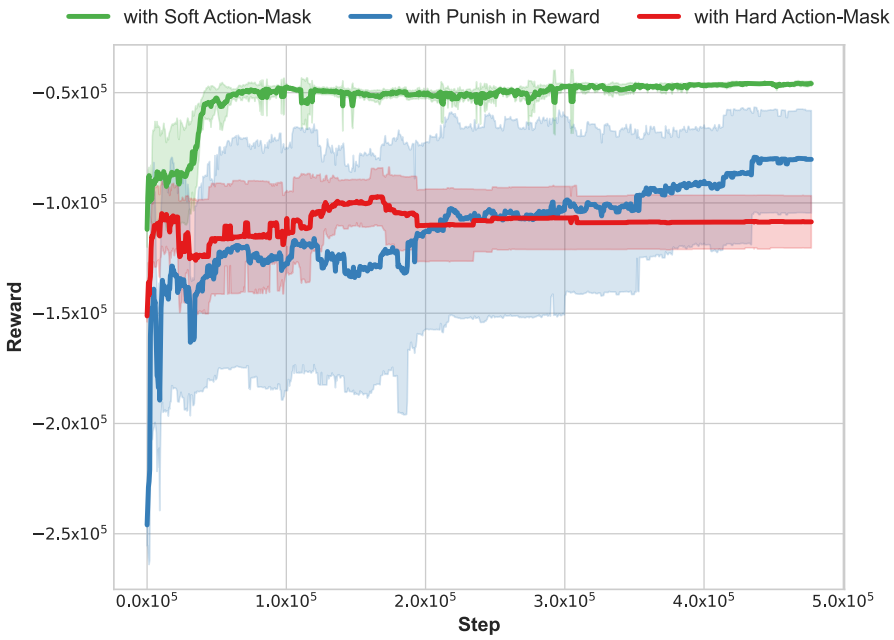


Figure 4. The record of historical mean reward and standard deviation of three RL method.

SAM and HAM have significantly higher initial rewards compared to purely learning by reward, as both HAM and SAM suppress adverse actions in a given state based on existing experience. The SAM trajectory in the figure has a slightly higher initial reward than the HAM, due to the fact that the SAM rules can be optimized during the learning

process, allowing for more leeway when initializing the masking rules. In addition, it can be seen from the late training period that the average reward level of HAM is surpassed by the pure reward learning method that is allowed to explore randomly. Although the pure reward method shows potential in the later stages, the the range shows that its efficiency and effectiveness are influenced by the initialized parameters and the randomness of sampling during training, which is difficult to accept in the early stages. SAM, on the other hand, is based on rule-based initialization and is expected to start sub-optimally with knowledge. Compared to learning from rewards alone, SAM achieves a significant improvement upfront and stabilizes at a superior level of performance. In addition, the exploration is based on a soft rule framework with a small standard deviation of rewards, which is more stable than the original approach.

3.2. Effect Comparison

The strategies obtained by the three reinforcement learning methods were validated on 30 days of test data and compared with the threshold control strategy. As the threshold control strategy has different control effects when different thresholds are selected, two sets of thresholds were implemented in the experiment for comparison. The results are shown in Figure 5, which compares the control effect in terms of the percentage of the total fan running time, the time percentage when the temperature and humidity are out of the appropriate range, and the percentage when the skylight is in frequent operation.

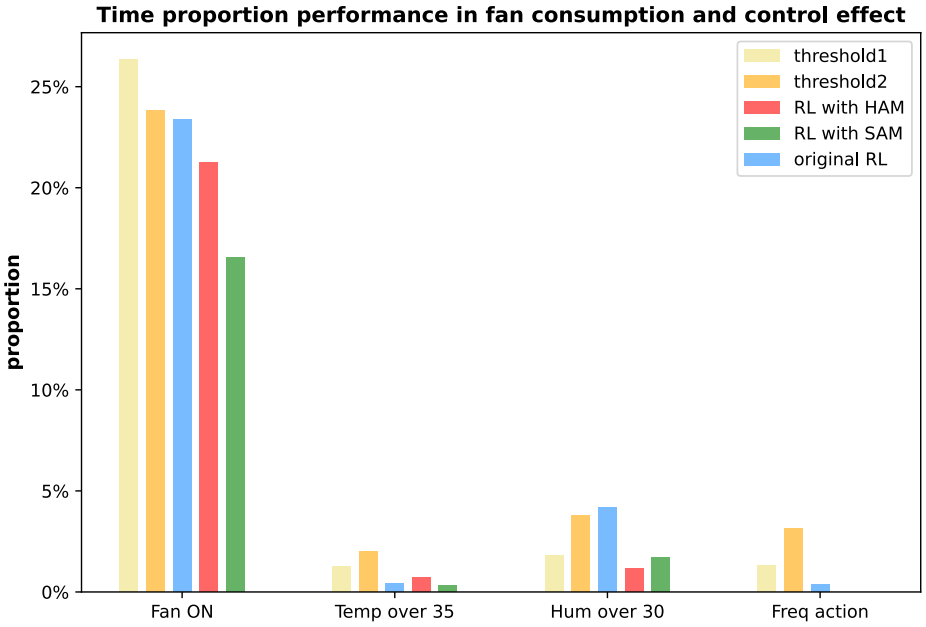


Figure 5. The control effect comparison between two threshold methods and policies trained by RL with HAM/SAM in terms of the percentage of the total running time of fans, the time percentage when the temperature and humidity are out of the appropriate range, and the percentage when the skylight is in frequent operation.

In the threshold control strategy with two sets of parameters, the more aggressive one for maintaining a suitable environment is less likely to result in frequent operation, but at the same time has a longer fan running time, accounting for 26.35% and 23.82% of the total control process time, respectively. The effect of adjusting the threshold parameters to reduce consumption is very limited and leads to frequent movements. Compared to the threshold control, the reinforcement learning method is effective in reducing the runtime, but the original reward RL cannot completely avoid frequent actions, which may be related to the reward weight setting as well as the limited generalization ability and low learning efficiency. With the addition of the Action Mask, frequent actions were completely avoided

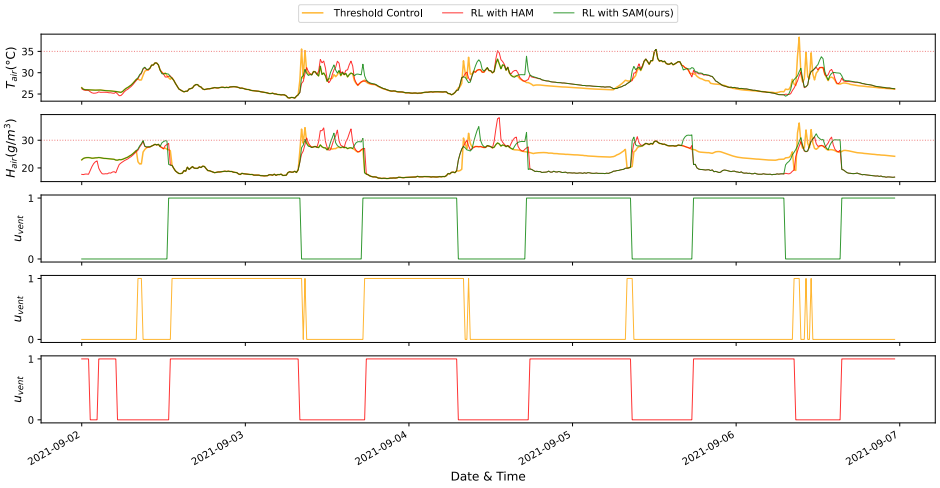
and unsuitable states were effectively reduced. The policy trained by SAM resulted in better temperature suitability. Although the frequency of out-of-range humidity was slightly higher than HAM, it remained at a lower level. What is more important is that the SAM fan runtime was only 16.56% of the total process, a reduction of 37.15% and 30.47% compared to the threshold control policy with two sets of parameters, and 22.07% compared to the HAM method with the same training rounds, significantly reducing the fan runtime while maintaining better environmental suitability.

3.3. Performance Comparison

3.3.1. Threshold Control vs. RL

Temperature and humidity controlled by threshold control, the HAM and SAM training policies, and the corresponding control output u_{pad} , u_{vent} over five days are shown and compared in Figure 6; the sampling period T_s is 900 s.

(**a**)



(**b**)

Figure 6. Comparison of the control effect in terms of temperature, humidity, the output of wet curtain fan and skylight. (**a**) Comparison of temperature, humidity, and the output of wet curtain fan for 5 days under the threshold control and policies trained by RL with HAM/SAM; T_s = 900 s. (**b**) Comparison of temperature, humidity, and the output of skylight for 5 days under the threshold control and policies trained by RL with HAM/SAM; T_s = 900 s.

It is clear that SAM and HAM effectively shorten the fan running time and that there are no frequent movements caused by improperly set threshold parameters, resulting in energy-saving optimization. In a few cases, the threshold control effect resulted in temperatures outside the appropriate range, due to large fluctuations in temperature caused by frequent actions, whereas the temperatures under both HAM and SAM policy control were maintained within the appropriate range. The study was conducted in a summer and autumn greenhouse. The data did not appear to be below the lower limit, so only the upper appropriate limit is plotted on the graphs. In fact, the temperatures under the HAM and SAM policies are sometimes slightly higher than those under the threshold control, since the regional reward is within the appropriate range for the factor, i.e., it penalizes mainly energy consumption, sacrificing some cooling effect in return for a reduction in fan operating hours. In a few cases, the RL method may exceed the upper limit of the suitable range because the temperature target in the reward is weighted higher than the humidity, while a short time spent out of the suitable range has less impact on the long-term benefits, and it quickly returns to the suitable area, which can be considered within the control target allowed.

3.3.2. HAM vs. SAM

SAM tends to have lower temperature peaks than HAM, slightly lower humidity curve exceedances than HAM, more stable control overall, and shorter fan run times compared to HAM. These benefits are considered to be mainly related to the trainable adjustment capability of SAM, where setting a masking rule in HAM prevents the exploration of policy under some condition state, whereas SAM can still learn a potentially better condition judgment than human experience. The second possible reason is SAM is more efficient than HAM, and it is more likely to learn better policies in the same amount of time.

In conclusion, through simulated environment experiments, it has been proven that the method provided has significant advantages over RL with pure rewards or HAM in terms of improving the levels of the initialization policy, learning efficiency, stability, and potential of optimization effect. Starting exploration from a safe and sub-optimal policy, our algorithm regulates the greenhouse environment with a lower initial cost, further reducing energy consumption and preventing facilities from running frequently due to environmental changes during the control process.

4. Conclusions

This work put forward a SAM net initialized on rules to start reinforcement learning to cope with the problem of energy conservation control in a Venlo-type greenhouse. It combines the interpretability of knowledge and rules with the strong learnability of a randomly initialized fully connected network, guiding exploration and ensuring the warm start of the policy, the safety of the training process, and the potential of policy exploration.

The advantages of the method proposed in this paper are summarized as follows:

Model-free dynamic optimization: Without model control objects, the agent dynamically optimizes policies through interaction with the environment and feedback on long-term benefits.

Safe Exploration: By adopting action masks, unsafe and unstable actions under specific conditions can be avoided. Moreover, energy consumption for the motor operation of the three-state actuator can be indirectly reduced.

Exploration guided by human knowledge: Random exploration in the early stage of learning results in a high cost of trial-and-error to the actual greenhouse application. The proposed approach embeds part of human knowledge into the randomly initialized policy network, ensuring that training starts from a higher level, and the learning process is constrained by the manually initialized rule framework, which effectively guides the stable and safe learning of agents.

Trainable networked rules of masks: Considering the difficulties in establishing complete rules in the greenhouse and determining rules that are disadvantageous for in-depth

exploration, the proposed method constructs networked rules on the basis of fuzzy knowledge, and the initialized parameters can be updated by the policy gradient algorithm during the learning process, so as to intelligently adjust more appropriate rules. This approach combines the interpretability of rules and the learnability of neural networks.

The proposed method is one of generalization, especially in large, precisely controlled greenhouses, where energy-saving optimization has greater value for improving economic benefits. In addition, the rich data available can provide more feedback information, which greatly benefits control based on reinforcement learning. As a consequence, this method is worthy of further in-depth study and has a long-term significance in improving the economic benefits of modern agricultural facilities.

Author Contributions: Conceptualization, L.C., L.X. and R.W.; data curation, L.C.; methodology, L.C.; software, L.C.; validation, L.C.; formal analysis, L.C.; investigation, L.C.; resources, L.X.; writing—original draft preparation, L.C.; writing—review and editing, L.X. and R.W.; visualization, L.C.; supervision, L.X. and R.W.; project administration, L.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Shanghai Municipal Science and Technology Commission Innovation Action Plan (Grant No. 20dz1203800), the National Natural Science Foundation of China (Grant No. 61973337), and the US National Science Foundation's BEACON Center for the Study of Evolution in Action (#DBI-0939454).

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

RL	Reinforcement Learning
HAM	Hard Action Mask
SAM	Soft Action Mask

References

1. Xu, L.; Wu, J. *An Algorithm of Greenhouse Multi Factors Coordination*; Technical Report; China National Patent Office: Beijing, China, 2007.
2. Wang, Y. Study of Light Supplementation Strategy for Multi-Factor Coupled Greenhouse Environments. Master's Thesis, Tongji University, Shanghai, China, 2018.
3. Chen, S. Study of Control System Solution and Algorithm Implementation for Greenhouse Microclimate. Master's Thesis, Tongji University, Shanghai, China, 2019.
4. Van Beveren, P.; Bontsema, J.; Van Straten, G.; Van Henten, E. Minimal heating and cooling in a modern rose greenhouse. *Appl. Energy* **2015**, *137*, 97–109. [CrossRef]
5. Xu, L.; Hu, Q.; Hu, H.; Goodman, E. Conflicting multi-objective compatible optimization control. In *New Achievements in Evolutionary Computation*; INTECH: Rijeka, Croatia, 2010; pp. 113–134.
6. Hu, H.; Xu, L.; Wei, R.; Zhu, B. Multi-objective control optimization for greenhouse environment using evolutionary algorithms. *Sensors* **2011**, *11*, 5792–5807. [CrossRef] [PubMed]
7. Piñón, S.; Camacho, E.; Kuchen, B.; Peña, M. Constrained predictive control of a greenhouse. *Comput. Electron. Agric.* **2005**, *49*, 317–329. [CrossRef]
8. Qin, L.; Ma, G.; Chu, Z.; Wu, G. Modeling and control of greenhouse temperature-humidity system based on grey prediction model. *Trans. Chin. Soc. Agric. Eng.* **2016**, *32*, 233–241.
9. Xu, L.; Su, Y.; Liang, Y. Requirement and current situation of control-oriented microclimate environmental model in greenhouse system. *Trans. Chin. Soc. Agric. Eng.* **2013**, *29*, 1–15.
10. Taki, M.; Ajabshirchi, Y.; Ranjbar, S.F.; Rohani, A.; Matloobi, M. Heat transfer and MLP neural network models to predict inside environment variables and energy lost in a semi-solar greenhouse. *Energy Build.* **2016**, *110*, 314–329. [CrossRef]
11. Nabavi-Pelesaraei, A.; Abdi, R.; Rafiee, S. Neural network modeling of energy use and greenhouse gas emissions of watermelon production systems. *J. Saudi Soc. Agric. Sci.* **2016**, *15*, 38–47. [CrossRef]

12. Kavga, A.; Kappatos, V. Estimation of the temperatures in an experimental infrared heated greenhouse using Neural Network models. *Int. J. Agric. Environ. Inf. Syst. (IJAEIS)* **2013**, *4*, 14–22. [CrossRef]
13. Francik, S.; Kurpaska, S. The use of artificial neural networks for forecasting of air temperature inside a heated foil tunnel. *Sensors* **2020**, *20*, 652. [CrossRef]
14. Tchamitchian, M.; Kittas, C.; Bartzanas, T.; Lykas, C. Daily temperature optimisation in greenhouse by reinforcement learning. *IFAC Proc. Vol.* **2005**, *38*, 131–136. [CrossRef]
15. Ban, B.; Kim, S. Control of nonlinear, complex and black-boxed greenhouse system with reinforcement learning. In Proceedings of the 2017 International Conference on Information and Communication Technology Convergence (ICTC), Jeju Island, Republic of Korea, 18–20 October 2017; pp. 913–918.
16. Wang, L.; He, X.; Luo, D. Deep reinforcement learning for greenhouse climate control. In Proceedings of the 2020 IEEE International Conference on Knowledge Graph (ICKG), Nanjing, China, 9–11 August 2020; pp. 474–480.
17. Somov, A.; Shadrin, D.; Fastovets, I.; Nikitin, A.; Matveev, S.; Hrinchuk, O. Pervasive agriculture: IoT-enabled greenhouse for plant growth control. *IEEE Pervasive Comput.* **2018**, *17*, 65–75. [CrossRef]
18. Afzali, S.; Mosharafian, S.; van Iersel, M.W.; Velni, J.M. Optimal Lighting Control in Greenhouses Equipped with High-intensity Discharge Lamps Using Reinforcement Learning. In Proceedings of the 2021 American Control Conference (ACC), New Orleans, LA, USA, 25–28 May 2021; pp. 1414–1419.
19. Fujimoto, S.; Meger, D.; Precup, D. Off-policy deep reinforcement learning without exploration. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 2052–2062.
20. Fujimoto, S.; Conti, E.; Ghavamzadeh, M.; Pineau, J. Benchmarking batch deep reinforcement learning algorithms. *arXiv* **2019**, arXiv:1910.01708.
21. Kumar, A.; Fu, J.; Soh, M.; Tucker, G.; Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *Adv. Neural Inf. Process. Syst.* **2019**, *32*.
22. Wu, Y.; Tucker, G.; Nachum, O. Behavior regularized offline reinforcement learning. *arXiv* **2019**, arXiv:1911.11361.
23. Kumar, A.; Zhou, A.; Tucker, G.; Levine, S. Conservative q-learning for offline reinforcement learning. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 1179–1191.
24. Siegel, N.Y.; Springenberg, J.T.; Berkenkamp, F.; Abdolmaleki, A.; Neunert, M.; Lampe, T.; Hafner, R.; Heess, N.; Riedmiller, M. Keep doing what worked: Behavioral modelling priors for offline reinforcement learning. *arXiv* **2020**, arXiv:2002.08396.
25. Liu, Z.; Cen, Z.; Isenbaev, V.; Liu, W.; Wu, S.; Li, B.; Zhao, D. Constrained variational policy optimization for safe reinforcement learning. In Proceedings of the International Conference on Machine Learning, Baltimore, MD, USA, 17–23 July 2022; pp. 13644–13668.
26. Zhang, W.; Cao, X.; Yao, Y.; An, Z.; Xiao, X.; Luo, D. Robust Model-based Reinforcement Learning for Autonomous Greenhouse Control. In Proceedings of The 13th Asian Conference on Machine Learning (ACML), Virtual conference, 17–19 November 2021; pp. 1208–1223.
27. Berkenkamp, F.; Turchetta, M.; Schoellig, A.; Krause, A. Safe model-based reinforcement learning with stability guarantees. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
28. Hu, Z.; Ma, X.; Liu, Z.; Hovy, E.; Xing, E. Harnessing deep neural networks with logic rules. *arXiv* **2016**, arXiv:1603.06318.
29. Okajima, Y.; Sadamasa, K. Deep neural networks constrained by decision rules. In Proceedings of the AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January 2019; Volume 33, pp. 2496–2505.
30. Silva, A.; Gombolay, M. Encoding human domain knowledge to warm start reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; Volume 35, pp. 5042–5050.
31. Su, Y.; Xu, L.; Goodman, E.D. Greenhouse climate fuzzy adaptive control considering energy saving. *Int. J. Control. Autom. Syst.* **2017**, *15*, 1936–1948. [CrossRef]
32. Vanthoor, B.H. *A Model-Based Greenhouse Design Method*; Wageningen University and Research: Wageningen, The Netherlands, 2011.
33. Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Dębiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. Dota 2 with large scale deep reinforcement learning. *arXiv* **2019**, arXiv:1912.06680.
34. Ye, D.; Liu, Z.; Sun, M.; Shi, B.; Zhao, P.; Wu, H.; Yu, H.; Yang, S.; Wu, X.; Guo, Q.; et al. Mastering complex control in moba games with deep reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 6672–6679.
35. Huang, S.; Ontañón, S. A closer look at invalid action masking in policy gradient algorithms. *arXiv* **2020**, arXiv:2006.14171.
36. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
37. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.