A Simulator-based Planning Framework for Optimizing Autonomous Greenhouse Control Strategy

Zhicheng An*[1], Xiaoyan Cao*[2], Yao Yao*[1], Wanpeng Zhang[3], Lanqing Li[4], Yue Wang[3], Shihui Guo[2] and Dijun Luo[4]

[1] Tsinghua-Berkeley Shenzhen Institute, Tsinghua University
[2] School of Informatics, Xiamen University
[3] Tsinghua University
[4] Tencent AI Lab
{azc19, y-yao19, zhangwp19}@mails.tsinghua.edu.cn, wangyue@mail.tsinghua.edu.cn
holmescaoxy@gmail.com, guoshihui@xmu.edu.cn
{lanqingli, dijunluo}@tencent.com

Abstract

The rapidly growing global population presents challenges and demands for efficient production of healthy fresh food. Autonomous greenhouses equipped with standard sensors and actuators (such as heating and lighting), which enable control of indoor climate for crop production, contribute to producing higher yields. However, they require skilled and expensive labor, as well as a large amount of energy. An autonomous greenhouse control strategy, powered by AI algorithms by optimizing the yields and resource use simultaneously, offers an ideal solution to the dilemma. In this paper, we propose a two-stage planning framework to automatically optimize greenhouse control setpoints given specific outside weather conditions. First, we take advantage of cumulative planting data and horticulture knowledge to build a multi-modular simulator, using neural networks to simulate climate change and crop growth in the greenhouse. Second, two AI algorithms (reinforcement learning and heuristic algorithm) are applied as planning methods to obtain optimal control strategies based on the simulator. We evaluate our framework on a cherry tomato planting dataset and demonstrate how the simulator is able to simulate greenhouse planting processes with high accuracy and fast speed. Moreover, the control strategies produced by the AI algorithms all obtain superhuman performance, and in particular, significantly outperform all teams of the second "Autonomous Greenhouse Challenge" in terms of net profits.

Introduction

The global population is growing rapidly and is accompanied by the increasing demand for healthy and fresh food. Greenhouses and indoor farming systems have the potential to maintain suitable conditions for crop growth regardless of outdoor season and climate so as to prolifically and stably supply vegetables (Graamans et al. 2018). In many countries, the greenhouse industry is becoming an important crop production sector (CBI 2019). Meanwhile, the scarcity of natural resources is a rising problem around the world, which makes improving the utilization efficiency of energy an urgent matter (Bank 2020). Therefore, a greenhouse grower needs to balance yields and resource consumption. However, laborers qualified for operations in high-tech greenhouses (Sparks 2018) are expensive and scarce. Consequently, as farms expand, overseeing multiple greenhouse compartments simultaneously becomes a serious challenge.

Modern high-tech greenhouses are equipped with standard sensors, actuators (such as heating, lighting, CO_2 dosing, etc.) and process computers in order to provide a suitable climate for crop growth. A greenhouse grower needs to determine their climate and irrigation strategy based on experience and determine control setpoints manually. The process computers are responsible for controlling the execution of the actuators' setpoints. Sensors collect signals from indoor climate and crop status to provide feedback for future decision-making. Generally, in a typical 160-day cycle of crop growing, the dimensions of space for controlling parameters alone are astronomical. Humans, even greenhouse experts, are only able to give coarse-grained control strategies based on extremely partial observations.

Indeed, determining the control strategy for a greenhouse is an extremely complex planning problem. With a faithful simulator, we can utilize AI algorithms to assist in finding the optimal solution. In the last several decades, dynamic climate models have been widely developed (Cate 1984; Tantau 1980; Van Straten et al. 2010), and there are also works targeting crop models (Gary, Jones, and Tchamitchian 1998; Marcelis et al. 2009). However, existing models rely mainly on expert knowledge (physics and biology), which are difficult to transfer to other conditions. Even worse, the inference speed of these models, based on expert rules, is slow. Today, neural networks, or deep learning, have replaced the expert system in many fields (Chellapilla et al. 1999; Base 1995; Imrak 2008), but their application in greenhouse climate and crop simulation is lacking.

Deep reinforcement learning methods that have emerged

*These authors contribute equally to this work.

in recent years (Van Hasselt, Guez, and Silver 2015; Schulman et al. 2017; Haarnoja et al. 2018; Fujimoto, Meger, and Precup 2019), and mature heuristic algorithms (Kuo, Cheng, and Chen 2003; Ponsich et al. 2008) are usually used to solve planning problems in practice applications. There are some works using reinforcement learning algorithms to control greenhouse climate automatically in order to use resources more efficiently (Bhattacharya, Lobbrecht, and Solomatine 2003; Wang, He, and Luo 2020). To the best of our knowledge, there has been little progress in simulator-based AI to optimize yields and resource consumption simultaneously.

This paper aims to resolve the aforementioned challenges and offer a solution to obtain optimal greenhouse control strategy automatically given specific outside weather. First, based on neural networks and horticultural knowledge, we build a greenhouse climate and crop simulator in a three-layer cascade. Second, we adopt state-of-the-art deep reinforcement learning (Haarnoja et al. 2018) and a mature heuristic algorithm (Bhandari, Murthy, and Pal 1996) as auxiliary tools, optimizing greenhouse control strategies based on our simulator. The contributions of our work include:

- We propose a novel two-stage framework to obtain optimal greenhouse control strategy automatically. In the first stage, we build a greenhouse climate and crop simulator; in the second stage, we use two AI algorithms to optimize control strategies based on our simulator.
- We design a data-driven multi-modular neural-network simulator based on the cherry tomato planting dataset, which can simulate greenhouse climate and crop states with high accuracy and fast speed on an hourly basis.
- Experiments on our simulator demonstrate that the control strategies offered by AI algorithms significantly outperform all teams of the second "Autonomous Greenhouse Challenge" (Hemming et al. 2020) in terms of net profits.

Problem Statement

In this work, we formulate the greenhouse control problem as a deterministic Markov decision process (MDP) problem. In the following, we give the necessary background for the MDP and formally introduce the problem of autonomous greenhouse control under weather uncertainty.

MDP Introduction

A MDP can be denoted by a tuple $\mathbb{S} = (\mathcal{S}, s_0, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$, where $\mathcal{S} \subseteq \mathbb{R}^n$ and $\mathcal{A} \subseteq \mathbb{R}^m$ represent the state and action spaces, respectively, $s_0 \in \mathcal{S}$ is the initial state, $\mathcal{P}(s' \mid s, a)$ denotes the probability that $s' \in \mathcal{S}$ is reached after executing action $a \in A$ in state $s \in \mathcal{S}$, $\mathcal{R}(s, a, s')$ is the immediate reward of executing action a to transition from state s to s', and $\gamma \in (0, 1]$ is the discount factor to encourage short-term gains. A solution to $\mathbb{S}$ is a *stochastic* policy π that maps a state s to a probability distribution over $\mathcal{A}$. We denote by $\pi(a \mid s)$ the probability of a to be selected facing state s. At each time step t, the agent is asked to select an action a_t according to policy π, generating a trajectory of states, actions and rewards, i.e., $\tau = (s_0, a_0, r_0, s_1, a_1, \ldots, s_T)$, where $a_t \sim \pi(. \mid s_t)$, $s_{t+1} \sim \mathcal{P}(. \mid s_t, a_t)$, $r_t = \mathcal{R}(s_t, a_t, s_{t+1})$, and T is the terminate time step. For a given MDP, the optimal policy π^* maximizes the cumulative expected return:

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, E_\tau \left[\sum_{t=0}^{T-1} \gamma^t r_t \right] \tag{1}$$

Greenhouse Control in Uncertain Weather

The goal in optimizing autonomous greenhouse control strategy under non-deterministr weather is to find a policy that maximizes the cumulative expected return and considers crop yields and resource consumption simultaneously.

There are many factors that affect the greenhouse planting process, but we focus only on the factors that have a large impact in order to simplify the problem. Formally, a *greenhouse control process* is a tuple $G =< C, G, P, A, W >$, where $c \in C \subseteq \mathbb{R}^{k_1}$ denotes the indoor climate of the greenhouse, $g \in G \subseteq \mathbb{R}^{k_2}$ is the crop growth state, $p \in P \subseteq \mathbb{R}^+$ is the current production, $a \in A$ follows the control strategy and $w \in W$ represents the outside weather.

In autonomous greenhouse control, the indoor climate c, adjusted by action a, greatly affects the crop state g and production p. Their relationship of influence with each other is subject to specific physical laws, so it is deterministic. In contrast, outside weather w is independent and uncertain. We assume that there exists a set of weather W, and the probability of observing outside weather $w \in W$ at time t is given by the weather model $P_w(w \mid t)$. In principle, weather models vary from place to place, raising challenges to the optimization of automatic greenhouse control.

Deterministic Autonomous Greenhouse Control

In our implementation, the weather data available to us is specified. Under this condition, the greenhouse control problem can be modeled as a deterministic MDP, whose transition function $\mathcal{P}(s, a)$ is deterministic rather than a probability distribution. Additionally, we assume that other variables beyond the variables in Table 1 and Table 2 have no effect on the state-transition function. We are now able to specify the state space, action space, reward function, and transition function of this deterministic MDP.

- State space: For each time step t, the state consists of a four-tuple $< w, c, g, p >$. The specific variables are given in Table 1. Each variable of the state is a real value with different units which encode the different information, determining the growth status of the crop.
- Action space: We focus on the actions, i.e., temperature, CO_2 concentration, lighting, and irrigation as the four most important control variables of the greenhouse for learning automatic control strategy.
- Reward function: In greenhouse control, we aim to increase crop yields while decreasing costs associated with the control strategy. In the simplest trade-off, we define the reward as the gains minus the costs. The specific formulation of the reward function is given in the following section.
- Transition function: The transition function in autonomous greenhouse control has no explicit expression.

Category	Name	Description	Unit
Outside weather	Iglob	outside solar radiation	W/m^2
	Tout	outside temperature	$°C$
	RHout	outside humidity	%
	CO_2out	outside CO_2 concentration	*ppm*
	Windsp	outside wind speed	m/s
	Tsky	virtual sky temperature	$°C$
Indoor climate	AirT	inside temperature	$°C$
	AirRH	inside humidity	%
	AirCO_2	inside CO_2 concentration	*ppm*
	PAR	light intensity above crop	$umol/s$
Crop state	LAI	leaf area index	-
	PlantLoad	number of growing fruits	$fruits/m2$
	NetGrowth	photosynthesis net growth	$kg\, CH_2O/s$
Production	FW	fruit fresh weight	kg/m^2

Table 1: Basic state variables defined in this paper.

Action (setpoint)	Unit
greenhouse temperature setpoint	$°C$
greenhouse CO_2 concentration	*ppm*
light on/off	-
irrigation on/off	-

Table 2: Basic action variables defined in this paper.

In this work, we focus on estimating and approximating this function through a multi-modular model.

Problem Formulation

Given the deterministic autonomous greenhouse control MDP, the optimization objective is defined as:

$$\underset{\pi}{\operatorname{argmax}} \sum_{t=0}^{T-1} \gamma^t r_t$$
$$s.t. \quad \begin{cases} \gamma = 1, \\ a_t = \pi(s_t) & (i = 0, \cdots, T-1), \\ s_{t+1} = \mathcal{P}(s_t, a_t) & (i = 0, \cdots, T-1), \\ r_t = \mathcal{R}(s_t, a_t, s_{t+1}) & (i = 0, \cdots, T-1), \end{cases}$$

where the reward function $\mathcal{R}$ and the transition function P will be introduced in the following.

Simulator as an Approximation

In the preliminaries, we model the optimization of the greenhouse control strategy as a single objective optimization problem, where the ground truth of the real transition function $\mathcal{P}$ in the greenhouse is implicit. We can build a simulator to approximate and fit this transition function. However, the construction of a specific greenhouse planting simulator usually requires corresponding expert experience and physical models (Van Straten et al. 2010; Gary, Jones, and Tchamitchian 1998; Marcelis et al. 2009), and different domain knowledge is required for different crops, which limits the flexibility of building a simulator. In order to alleviate these problems, we divided the state transition function into three modules trained in a data-driven manner.

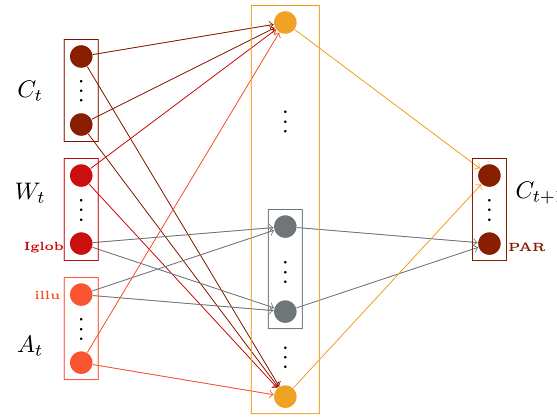

Figure 1: Greenhouse climate simulation module using current outside weather w_t, indoor climate c_t and control actions a_t to predict the next indoor climate c_{t+1}. PAR is determined by Iglob and illumination using a separate neuron channel.

Greenhouse Climate Simulation

The crop status is mainly determined by the indoor climate. Therefore, we first built a model to simulate greenhouse climate change. The indoor climate at the time step $t+1$, denoted by c_{t+1}, is influenced by current outside weather w_t and prior indoor climate c_t. Additionally, we applied action a_t to greenhouse actuators to adjust the indoor climate.

To sum up, the indoor climate change can be formalized as $c_{t+1} \leftarrow \mathbb{C}(w_t, c_t, a_t)$, where $\mathbb{C}$ represents the transition function of indoor climate change. Specifically, the dynamic greenhouse climate model is a tuple $< W, C, A >$: $W \times C \times A \rightarrow C$, where $W \subseteq \mathbb{R}^6$, $C \subseteq \mathbb{R}^4$, $A \subseteq \mathbb{R}^4$ (see Table 1 and Table 2 for details). Previous research mainly adopts aerodynamic methods for modeling $\mathbb{C}$, relying heavily on domain knowledge (Van Straten et al. 2010). In contrast, we propose a greenhouse climate simulation model $c_{t+1} \leftarrow \mathbb{C}_{\Theta_1}(w_t, c_t, a_t)$ based on neural networks, where Θ_1 represents network parameters. The model structure is shown in Figure 1.

Notice that the photosynthetically active radiation (PAR) is determined by solar radiation intensity and the power of the lamps (Alados, Foyo-Moreno, and Alados-Arboledas 1996); we designed a dedicated neuron channel for it (shown in Figure 1). Moreover, we assume the state of the greenhouse climate will change per hour, which is considered accurate enough to approximate reality. We adopt mean-square error as loss function; the formula is as follows:

$$\mathcal{L}(\Theta_1) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{4} \left(\hat{c}_i^{(k)} - c_i^{(k)} \right)^2 \tag{2}$$

where $\mathcal{L}(\Theta_1)$ represents the average loss of N samples, $\hat{c}_i^{(k)}$ and $c_i^{(k)}$ represent the real and prediction values of the k-th indoor climate variable of the i-th sample, respectively.

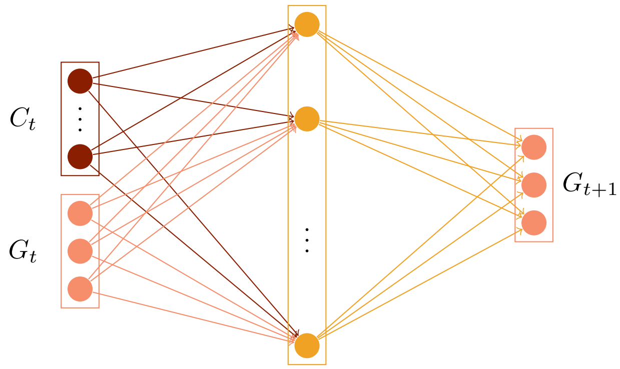

Figure 2: Growth state simulation module using current indoor climate c_t and crop growth state g_t to predict the next growth state g_{t+1}.

Crop Simulation

Different from indoor climate change, the final crop production is affected by indoor climate, crop state and initial crop production. Therefore, the crop production can be formulated as a tuple $< G, C, P >: G \times C \times P \rightarrow P$, where $G \subseteq \mathbb{R}^3$, $C \subseteq \mathbb{R}^4$ and $P \subseteq \mathbb{R}$. Detailed description of each element is shown in Table 1. The transition function of crop growth is modeled as a two-step function. First, we model the effect of the greenhouse climate on crop status, then model the relation of crop status and crop yield, corresponding to growth state module and production module.

Growth State Module. The state of greenhouse crops at time t is denoted by g_t, and the greenhouse indoor climate c_t can be calculated via the greenhouse climate model. We consider the direct effect of indoor climate on crop growth state; then, the growth state model can be formulated by a tuple $< G, C >: G \times C \rightarrow G$, where $G \in R^3$ and $C \in R^4$.

To get rid of dependence on horticultural knowledge (Gary, Jones, and Tchamitchian 1998), we built a neural network model $\mathbb{G}_{\Theta_2}$ to simulate the transition function $\mathbb{G}$ with respect to the crop growth process. Moreover, we used maximum and minimum normalization methods (Sola and Sevilla 1997) to eliminate dimensional differences. As the greenhouse climate changes, the crop status will be affected. Thus, we assumed that the crop status will change per hour just like the greenhouse climate status. Similarly to the greenhouse climate model, we used the mean-square error as the loss function.

Production Module. The change of crop production within one hour is intangible, so it is reasonable to assume yields will change only once per day. Since G corresponds to hour level and P corresponds to day level, we extend G to $\vec{G} = \left[G^{(0)}, G^{(1)}, \cdots, G^{(23)}\right]$ so that $\vec{G}$ is aligned with P on time scale, where $g_d^{(i)} \in G^{(i)}$ represents the growth state with respect to the i-th hour of day d. Due to crop growth state accumulating over time in our model, we only need to consider $g_d^{(23)}$. Then, the production p_{d+1} of day $(d+1)$ can be deduced by $p_{d+1} \leftarrow \mathbb{P}(p_d, g_d^{(23)})$, where $\mathbb{P}$ represents the underlying transition function of yield.

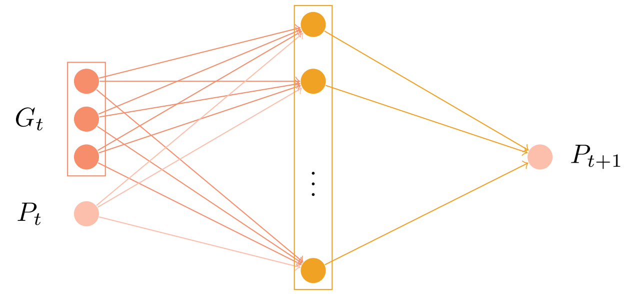

Figure 3: Production simulation module using current growth state g_t and production p_t to predict the next production p_{t+1}.

In a similar fashion to simulating greenhouse climate change, neural networks, parameterized by Θ_3, are used to model the crop production process: $P_{d+1} \leftarrow \mathbb{P}_{\Theta_3}(P_d, G_d^{(23)})$. The network structure is shown in Figure 3.

Reward Function

The organizers of the second "Autonomous Greenhouse Challenge" have publicized economic evaluation metrics. Similarly, we use $NetProfit$ as a reward, which depends on gains and resource consumption.[1] Note that we do not consider $FixedCosts$, which are constant regardless of the growing strategy.

Solution to Greenhouse MDP

There are many methods to solve MDP, such as dynamic planning methods and AI algorithms. AI algorithms have proven to be a more efficient way to solve MDP (Lee et al. 2020), so we consider two typical AI algorithms, including a reinforcement learning method and a heuristic algorithm. As shown in Figure 4, we propose a two-stage planning framework to solve the autonomous greenhouse control MDP problem. First, using cumulative data to train a greenhouse crop cultivation simulator built in the previous section. Second, according to the simulator, AI algorithms are applied as planning methods to obtain optimal control strategies, which will be described in the following.

Soft Actor-Critic Reinforcement learning (RL) algorithms have shown their effectiveness in solving autonomous decision-making problems. In this work, we choose Soft Actor-Critic (SAC) (Haarnoja et al. 2018) as a representative of RL to learn a control strategy based on the simulator. SAC is a state-of-the-art off-policy algorithm for continuous control problems, which is based on the maximum entropy RL framework. SAC is robustness to noise and encourages exploration by maximizing a weighted objective of the reward and the policy entropy (Lee et al. 2020). SAC alternates between a soft policy evaluation and a soft policy improvement to learn a critic, $Q_\theta(s, a)$, and a policy, $\pi_\phi(a|s)$.

[1]The detailed formula are available at https://www.kaggle.com/piantic/autonomous-greenhouse-challengeagc-2nd-2019?select=Economics.pdf

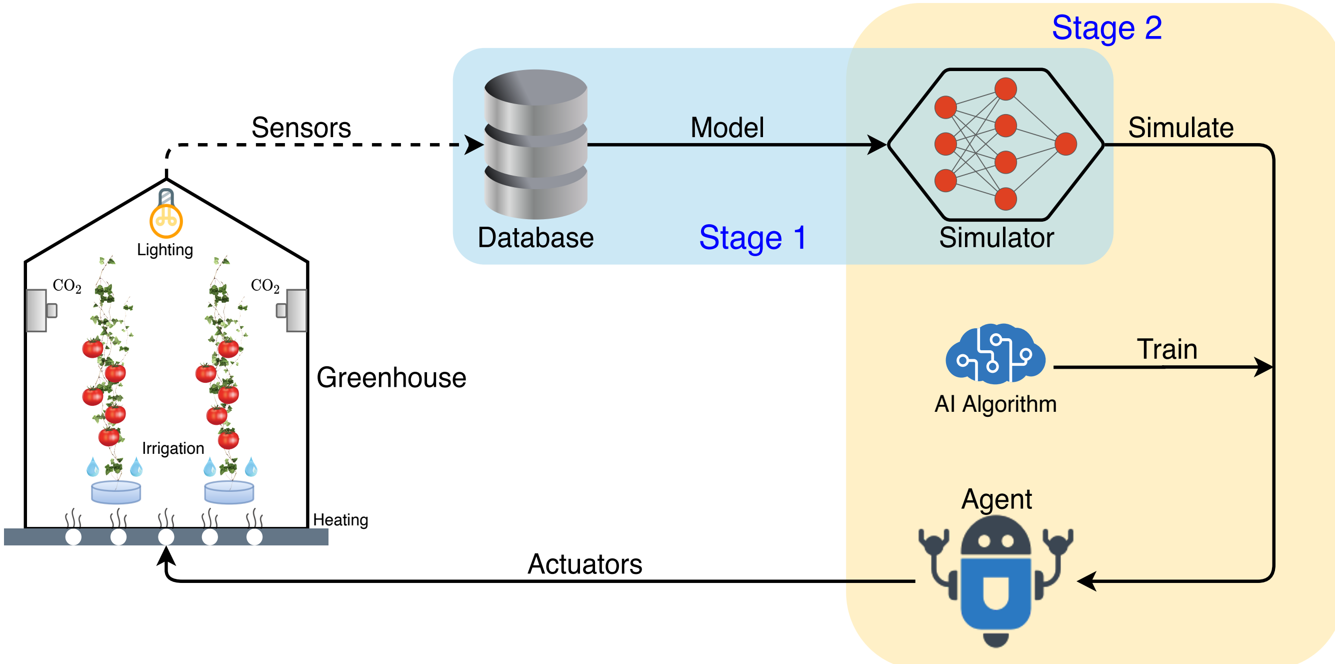

Figure 4: The general two-stage framework to optimize the autonomous greenhouse control strategy. To be specific, the optimizing process can be divided into two stages. First, we use the collected planting data in the real greenhouse as input to train a simulator that can approximate the real state transition function $\mathcal{P}$; then, we use AI algorithms to obtain the suboptimal policy in the simulator, which can be deployed as an approximate optimal control strategy to the real greenhouse.

At the soft policy evaluation step, a critic modeled as a neural network with parameters θ is optimized by minimizing the soft Bellman residual:

$$\mathcal{L}_{\text{critic}}^{\text{SAC}}(\theta) = \mathbb{E}_{\tau_t \sim \mathcal{B}}\left[\frac{1}{2}\left(Q_\theta(\mathbf{s}_t, \mathbf{a}_t) - \hat{Q}(\mathbf{s}_t, \mathbf{a}_t)\right)^2\right] \tag{3}$$

with

$$\hat{Q}(\mathbf{s}_t, \mathbf{a}_t) = r_t + \gamma \mathbb{E}_{a_{t+1} \sim \pi_\phi}\left[\hat{Q}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})\right] \tag{4}$$

Here, SAC uses a modified Q-function by adding an entropy-based regularization term to encourage the agent's exploration,

$$\hat{Q}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) = Q_{\bar{\theta}}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) - \alpha \log \pi_\phi(\mathbf{a}_{t+1} \mid \mathbf{s}_{t+1}) \tag{5}$$

where $\tau_t = (s_t, a_t, r_t, s_{t+1})$ is a transition sample generated by the simulator, $\mathcal{B}$ is a replay buffer storing the collected transitions, $\bar{\theta}$ are the delayed parameters and α is a temperature parameter. At the soft policy improvement step, the policy π with its parameters ϕ can be learned by directly minimizing the following loss:

$$\mathcal{L}_{\text{actor}}^{\text{SAC}}(\phi) = \mathbb{E}_{s_t \sim \mathcal{B}}\left[\mathcal{L}_\pi(\mathbf{s}_t, \phi)\right] \tag{6}$$

where

$$\mathcal{L}_\pi(\mathbf{s}_t, \phi) = \mathbb{E}_{a_t \sim \pi_\phi}\left[\alpha \log \pi_\phi(\mathbf{a}_t \mid \mathbf{s}_t) - Q_\theta(\mathbf{s}_t, \mathbf{a}_t)\right] \tag{7}$$

Here, the policy is modeled with a Gaussian distribution based on neural networks to handle continuous action spaces. SAC alternates between collecting transitions from interacting with the simulator and updating the function approximation using the stochastic gradients from batched samples from a replay buffer. The complete algorithm is described in Algorithm 1.

Algorithm 1: Soft Actor-Critic

```
Initialize policy π_φ and Q_θ.
for N_epoch iterations do
    for each step t do
        a_t ~ π_φ(a_t | s_t1)
        s_{t+1} ~ p(s_{t+1} | s_t, a_t)
        B ← B ∪ {(s_t, a_t, r(s_t, a_t), s_{t+1})}
    end
    for each gradient step do
        Update π_φ and Q_θ by minimising L_critic^SAC(θ)
          and L_critic^SAC(θ)
    end
end
```

Elitist Genetic Algorithm In the previous section, we model the greenhouse control strategy optimization problem as a deterministic NP-hard optimization problem, which could be solved by a heuristic algorithm (Michalewicz 2013; Ponsich et al. 2008) to find the suboptimal solution.

In this work, we apply a classic heuristic algorithm, elite retention genetic algorithm (EGA) (Rani, Suri, and Goyal

Algorithm 2: Elitist genetic algorithm

Input: Initial population size p_size, iteration limit itr_{limit}, and the composition of polysomy $polysomy$
Output: Optimal $strategy$ after itr_{limit} iterations

$pop \leftarrow$ InitializePopulation(p_size)
$strategy \leftarrow$ DecodeGenetics(pop)
$ObjV \leftarrow$ Simulation($strategy$)
$pop.FitnV \leftarrow$ FitnessEvaluation(pop, $ObjV$)
$parent \leftarrow pop$
for itr_{limit} *iterations* **do**
  $offspring \leftarrow$ RouletteSelect($parent$, p_size)
  for *each polysomy* c_i **do**
    $offspring \leftarrow$ Crossover($offspring$, c_i)
    $offspring \leftarrow$ Mutation($offspring$, c_i)
  end
  $pop \leftarrow parent \cup offspring$
  $strategy \leftarrow$ DecodeGenetics(pop)
  $ObjV \leftarrow$ Simulation($strategy$)
  $pop.FitnV \leftarrow$ FitnessEvaluation(pop, $Objv$)
  $parent \leftarrow$ SelectTopSizeFit(pop, p_size)
end
$BestIndividual \leftarrow$ SelectBestFit($parent$)
$strategy \leftarrow$ DecodeGenetics($BestIndividual$)
return $strategy$

2019) that can accelerate convergence in our scenario to obtain the optimal greenhouse control strategy. The procedure of EGA for greenhouse control strategy is summarized in Algorithm 2.

Considering that the state space of different variables is significantly different, for example, temperature and carbon dioxide concentration are continuous while illumination and irrigation are discrete, we adopt polychromosomal coding, in which the value of the same variable in a whole period shares one chromosome by concatenation.

Here, we use a binary-coded method and perform crossover and mutation on the binary string. Since the simulator requires real numbers as input, each individual in the population will be decoded before simulating. The simulator will return the cumulative rewards of the complete growing period as the objective value of each individual in the population.

Experiments

Dataset

We evaluate our framework on a tomato dataset collected from a reliable simulator built by Wageningen University & Research (WUR). This dataset includes more than 1,000 planting trajectories with adequate recordings of outside weather, indoor climate, crop state, production and cost. All state variables and action variables can be found in this dataset. Each trajectory in this dataset records hourly data of a 160-day planting episode.

Simulator Analysis

We build a test set by randomly selecting 50 planting trajectories from the cherry tomato dataset. Then we use 100, 500 and 1,000 planting trajectories randomly selected from the remaining dataset to train different simulators.

We use goodness of fit (R^2) (Anderson and Darling 1954) as the evaluation index of simulator accuracy. The test results of each simulator are shown in Table 3.

According to Table 3, it is evident that a simulator without prior horticultural knowledge does not have any generalization, which indicates that the state transition probability $\mathcal{P}$ of climate and crop growth in a greenhouse is extremely complicated and cannot be approximated by data alone.

As for the multi-modular simulators designed in conjunction with the automatic greenhouse scenario, we find that R^2 will increase with the amount of training data. The simulator trained with 1,000 planting trajectories reaches a satisfactory approximation of $\mathcal{P}$.

Notice that the last production module of the simulator and its fitting performance are slightly better than the crop growth module. One of the reasons for this is that the simulation granularity of the production module is day-level. Furthermore, the production module relies only on the state of the last hour of the day of the crop growth module, which greatly reduces the difficulty of the characterization.

Economic Variable Analysis

We use the dataset collected from the second "Autonomous Greenhouse Challenge". Five teams use AI algorithms to remotely control a greenhouse to grow cherry tomatoes over an around 5-month period. A team of experienced growers also participates in the competition as references. This dataset[2] contains comprehensive recordings of various quantities during the competition, such as greenhouse temperature and CO_2 concentration.

WUR develops a tomato simulator that can simulate the planting process from the above competition more accurately than ours, but one simulation period takes about 10 minutes in comparison to our simulator, which takes only 4 seconds. We compare our multi-modular simulator with the WUR simulator to test the planting process of the 160-day competition. We plot the test results of the four teams in Figure 5.

According to Figure 5, we find that 1) The calculation method that we design for $Costs$ is accurate and stable; 2) $Gains$: Only the simulator trained on the dataset using 1,000 trajectories could simulate the harvest situation more accurately, but the yield is higher in the second half. In addition, the simulation effects of other versions of the simulator are unstable; 3) $Netprofit$: The simulator using 1000 trajectories has similar effects to the WUR simulator, and the other versions are not enough to simulate the real planting process.

Furthermore, we explore the reason why the harvest of our simulator is higher in the second half by analyzing the simulation of the greenhouse climate module and the crop growth

[2]The dataset is publicly available at https://doi.org/10.4121/uuid:88d22c60-21b3-4ea8-90db-20249a5be2a7

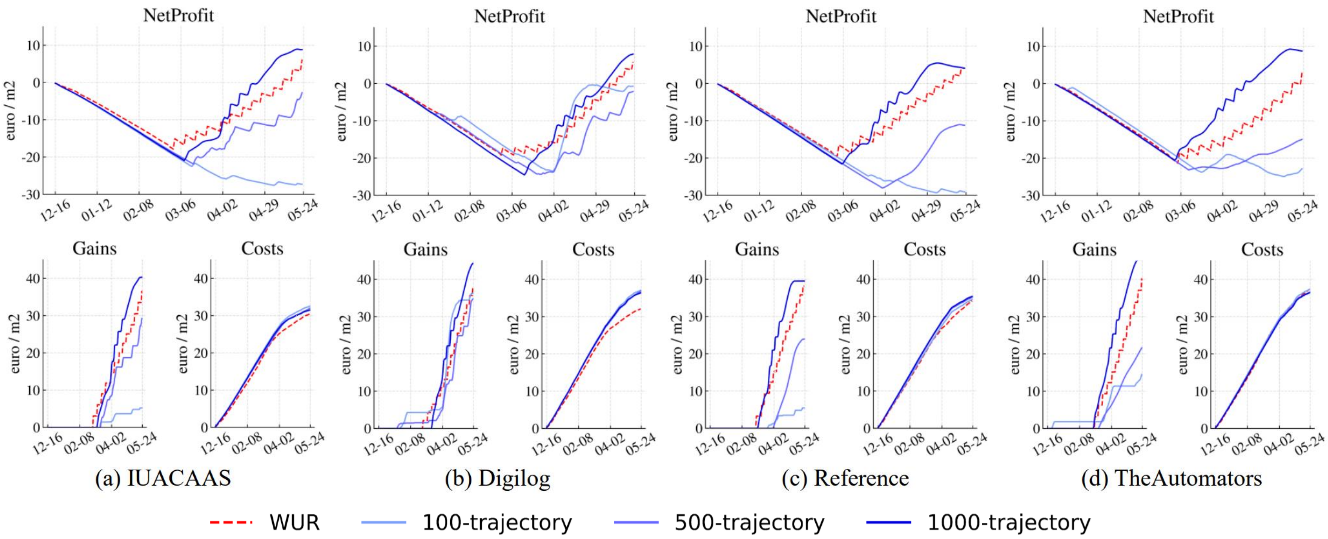

Figure 5: Results generated by simulators based on a dataset with different scales and a WUR simulator after applying the same strategy in each figure. (a)–(d) are 4 teams in the second "Autonomous Greenhouse Challenge".

Simulator	AirT	AirRH	AirCO$_2$	PAR	LAI	PlantLoad	NetGrowth	FW
no_prior	-72.320	-96.200	-437.526	-157.749	-4.903	-1920.908	-0.540	-16.455
100-trajectory	0.841	0.612	0.872	0.934	-0.135	0.383	0.227	0.278
500-trajectory	0.872	0.625	0.906	0.935	0.214	0.460	0.685	0.554
1000-trajectory	**0.961**	**0.851**	**0.966**	**0.935**	**0.630**	**0.829**	**0.959**	**0.952**

Table 3: Goodness of fit (R^2) of different variables in different simulators.

module. Figure 6 shows the comparison between the 1000-trajectory version of the simulator and the WUR tomato simulator after applying IUACAAS's policy.

Figure 6(a) shows that each hour's simulation deviation rate of every indoor climate variable in the greenhouse climate module is mostly within 10%, and the maximum error doesn't exceed 20%. Figure 6(b) is the simulation result of Leaf area index (LAI). Our simulator has a higher value in the first half, representing a stronger photosynthesis ability, which accumulates more $NetGrowth$ (Figure 6(d)). Thanks to better growth in the early stage, the fruit will mature and be picked earlier, that is, the $PlantLoad$ will decrease at a faster rate in the later stage (Figure 6(c)). Therefore, the output of our simulator will be higher in the later stage.

Comparison Methods

- **IUACAAS and Reference Policies** We took the IUACAAS (best performance policy in our simulator) team's and the Reference (human-expert policy) team's control strategy simulation results as baselines for our simulator.
- **EGA** EGA is an improved genetic algorithm that improves the convergence ability of the algorithm by adopting an elite retention strategy.
- **SAC** SAC is one of the state-of-the-art off-policy actor-critic RL algorithms for continuous control problems.

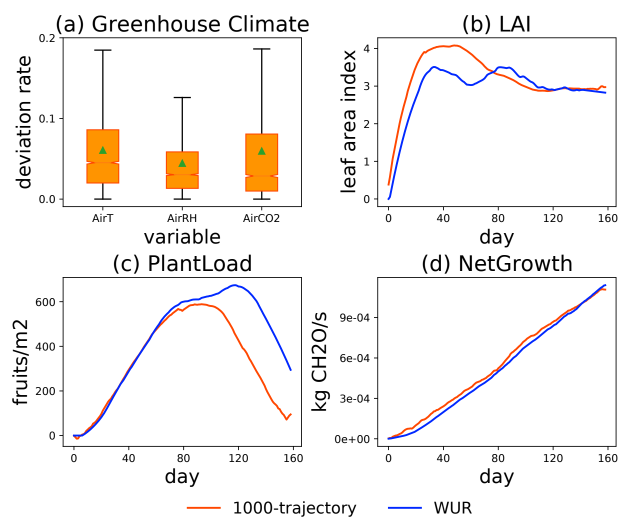

Figure 6: (a): Deviation rate distribution of variables between 1000-trajectory simulator and WUR simulator simulators in greenhouse climate module after applying IUACAAS's control strategy; (b)–(d): simulation values of LAI, $PlantLoad$ and $NetGrowth$ after applying IUACAAS's control strategy on two simulators.

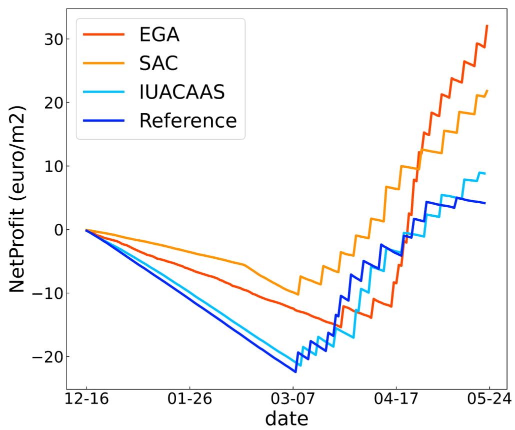

Figure 7: Cumulative balance of different control strategies.

Method	HeatCost	CO_2Cost	ElecCost	LaborCost
IUACAAS	6.550	**1.902**	17.714	5.271
Reference	8.018	2.531	19.524	5.271
SAC	**5.526**	2.296	**4.100**	5.271
EGA	6.230	2.123	12.302	5.271

Table 4: Different kinds of cost in the simulation via using our strategies and baselines.

Performance Comparison

As the WUR simulator is commercial equipment, it was open to us for a limited time (closed after the second "Autonomous Greenhouse Challenge"). For this reason, we can only conduct the experiments of comparing methods on our simulator. Table 4 and Figure 7 show the performance comparison of different methods on our simulator. We observe that: (1) The performance of each behavior policy is different, and IUACAAS outperforms the human-experts, who obtains a higher harvest with lower cost; (2) The net profit of the AI algorithms we apply greatly exceed the baselines. Taking SAC as an example, the AI algorithms have the ability to evaluate the benefit between the cost of resources consumed during each hour's action and the potential final harvest, thereby achieving fine control of the hourly granularity (shown in Figure 8) and simultaneously improving resource-use efficiency and maintaining high harvest. The baseline strategies could only achieve 3–6 adjustments in one day, and they remain almost unchanged for several days.

Convergence Analysis

Figure 9 plots the obtained reward in our simulator during the training of the policies. We find that both EGA and SAC can converge to a stable net profit, where EGA can achieve higher net profit. The possible reason for this is that each iteration of EGA will consider the impact of actions at each moment in the entire period on the final result. The SAC pays more attention to the benefit of decision-making in the present moment, while the importance of future benefit is gradually reduced. In our experiment, the planting cycle is nearly 4,000 hours, which makes it difficult for SAC to maximize the benefits of the final result, and thus, it falls into a locally optimal solution.

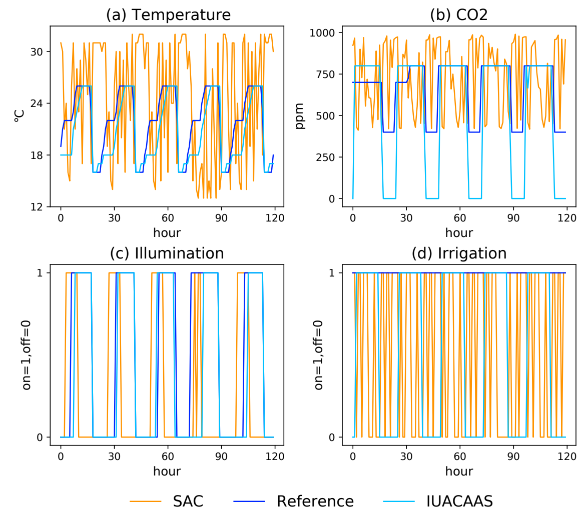

Figure 8: 120-hour control strategies of SAC, Reference and IUACAAS.

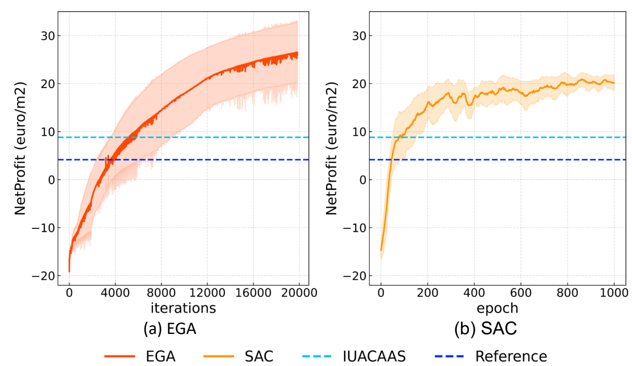

Figure 9: Training curves of SAC and EGA.

Conclusion

In this paper, we propose a two-stage planning framework to optimize the control strategy for autonomous greenhouse planting. First, we model the autonomous greenhouse control as a deterministic MDP problem and point out that the state transition function $\mathcal{P}$ is the key challenge. Then, we build a multi-modular simulator based on neural networks in a data-driven style to approximates $\mathcal{P}$. We use real data to verify that the simulator achieves good approximation to reality. On our simulator, two AI algorithms we used both significantly outperform all other teams' strategies of the second "Autonomous Greenhouse Challenge" in terms of net profits.

To further illustrate effectiveness of our framework in the real world, we have deployed this framework in real greenhouses based on IoT, cloud-native, and other technologies. The experiments are in progress, for which the results will be shown and analyzed in the subsequent works.

References

Alados, I.; Foyo-Moreno, I. y.; and Alados-Arboledas, L. 1996. Photosynthetically active radiation: Measurements and modelling. *Agricultural and Forest Meteorology* 78(1-2): 121–131.

Anderson, T. W.; and Darling, D. A. 1954. A test of goodness of fit. *Journal of the American statistical association* 49(268): 765–769.

Bank, W. 2020. Somalia country environmental analysis: diagnostic study on trends and threats for environmental and natural resources challenges. Technical report, World Bank, Washington, DC. URL https://openknowledge.worldbank.org/handle/10986/34058.

Base, L. T. 1995. *Neural Networks for Pattern Recognition*. Oxford University Press.

Bhandari, D.; Murthy, C.; and Pal, S. K. 1996. Genetic algorithm with elitist model and its convergence. *International Journal of Pattern Recognition and Artificial Intelligence* 10(06): 731–747.

Bhattacharya, B.; Lobbrecht, A.; and Solomatine, D. 2003. Neural networks and reinforcement learning in control of water systems. *Journal of Water Resources Planning and Management* 129(6): 458–465.

Cate, A. J. U. T. 1984. Modelling and (adaptive) control of greenhouse climates. *Netherlands Journal of Agricultural Science* 32(2).

CBI. 2019. Which trends offer opportunities or pose threads on the European fresh fruit and vegetables market? Available online: https://www.cbi.eu/market-information/fresh-fruit-vegetables/trends (accessed on 21 December 2020).

Chellapilla; Kumar; Fogel; and David, B. 1999. Evolving neural networks to play checkers without relying on expert knowledge. *IEEE Transactions on Neural Networks* .

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2052–2062.

Gary, C.; Jones, J. W.; and Tchamitchian, M. 1998. Crop modelling in horticulture: State of the art. *Scientia Horticulturae* 74(1-2): 3–20.

Graamans, L.; Baeza, E.; Van Den Dobbelsteen, A.; Tsafaras, I.; and Stanghellini, C. 2018. Plant factories versus greenhouses: Comparison of resource use efficiency. *Agricultural Systems* 160: 31–43.

Haarnoja, T.; Zhou, A.; Hartikainen, K.; Tucker, G.; Ha, S.; Tan, J.; Kumar, V.; Zhu, H.; Gupta, A.; Abbeel, P.; et al. 2018. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905* .

Hemming, S.; de Zwart, F.; Elings, A.; Petropoulou, A.; and Righini, I. 2020. Cherry tomato production in intelligent greenhouses—sensors and AI for control of climate, irrigation, crop yield, and quality. *Sensors* 20(22): 6430.

Imrak, C. E. 2008. Artificial neural networks application in duplex/triplex elevator group control system. *Strojniski Vestnik* 54(2): 103–114.

Kuo, J. T.; Cheng, W. C.; and Chen, L. 2003. Multi objective water resources systems analysis using genetic algorithms—application to Chou-Shui river basin Taiwan. *Water Science Technology* 48(10): 71–77.

Lee, K.; Laskin, M.; Srinivas, A.; and Abbeel, P. 2020. Sunrise: A simple unified framework for ensemble learning in deep reinforcement learning. *arXiv preprint arXiv:2007.04938* .

Marcelis, L. F. M.; Elings, A.; Visser, P. H. B. D.; Heuvelink, E.; Fischer, G.; Magnitskiy, S.; and Nicola, S. 2009. Simulating growth and development of tomato crop. In *International Symposium on Tomato in the Tropics*.

Michalewicz, Z. 2013. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer Science & Business Media.

Ponsich, A.; Azzaro-Pantel, C.; Domenech, S.; and Pibouleau, L. 2008. Constraint handling strategies in genetic algorithms application to optimal batch plant design. *Chemical Engineering Processing Process Intensification* 47(3): 420–434.

Rani, S.; Suri, B.; and Goyal, R. 2019. On the effectiveness of using elitist genetic algorithm in mutation testing. *Symmetry* 11(9): 1145.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* .

Sola, J.; and Sevilla, J. 1997. Importance of input data normalization for the application of neural networks to complex industrial problems. *IEEE Transactions on Nuclear Science* 44(3): 1464–1468.

Sparks, B. D. 2018. What is the current state of labor in the greenhouse industry? Available online: https://www.greenhousegrower.com/management/what-is-the-current-state-of-labor-in-the-greenhouse-industry/ (accessed on 21 March 2021).

Tantau, H. J. 1980. Climate control algorithms. *Acta Horticulturae* 49–57.

Van Hasselt, H.; Guez, A.; and Silver, D. 2015. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461* .

Van Straten, G.; Van Willigenburg, G.; Van Henten, E.; and Van Ooteghem, R. 2010. *Optimal Control of Greenhouse Cultivation*. CRC Press.

Wang, L.; He, X.; and Luo, D. 2020. Deep reinforcement learning for greenhouse climate control. In *IEEE International Conference on Knowledge Graph (ICKG)*, 474–480.