Learning Control for Air Conditioning Systems via Human Expressions

Qinglai Wei, *Member, IEEE*, Tao Li, and Derong Liu, *Fellow, IEEE*

***Abstract*—In this article, a deep reinforcement learning method is developed to solve air conditioning control problems through human expressions. The main contribution of this article is to design a deep reinforcement learning method for air conditioning control problems with human expressions as the input for the first time. The method aims to eliminate human sleepiness and improve people's work efficiency as much as possible. First, the air conditioning system and deep reinforcement learning methods are introduced. Second, the image processing algorithm for human expressions is described. Third, the deep Q-network method is designed to obtain the optimal control policy for air conditioning systems. Finally, simulation results are given to illustrate the present method that can effectively eliminate sleepiness and improve the work environment of people.**

***Index Terms*—Adaptive dynamic programming, air conditioning control, deep learning (DL), deep Q-network (DQN), human expressions, optimal control, reinforcement learning (RL), Q-learning.**

I. Introduction

AIR conditioning is the most popular tool to adjust the indoor temperature, which is always a key factor to affect people's work efficiency. In Hedge and Gaygen's research [1], it is proven that there is an optimum range of temperatures to make people feel comfortable while too high or too low temperature can cause people to lose productivity. Therefore, the control of indoor temperature is of great significance to improve the work efficiency of people. In previous research, many control methods for air conditioning systems have emerged, such as the classic proportional-integral-derivative control [2], the linear quadratic regulator control [3], the fuzzy control [4], and the model predictive control [5]. Previous control methods are mostly focused on guaranteeing the stability of the system [6], such as the indoor temperature. However, it is pointed out in [7] that control strategies stabilizing the indoor temperature cannot effectively improve people's work efficiency. Therefore, this article abandons the idea of ensuring temperature stability in previous studies and sets eliminating people's sleepiness as the main purpose. This article designs a self-learning controller using the image of human expressions as the input. It can learn the appropriate control strategy through self-learning, and then adjusts the temperature of the airflow blown by the air conditioning system to eliminate the human sleepiness.

Due to the complexity and uncertainty of the air conditioning system and the room thermal model, traditional model-based control methods often fail to achieve satisfactory results in practical applications. Therefore, many researches of applying machine learning algorithms to air conditioning systems have been conducted. Reinforcement learning (RL) is an area of machine learning that deals with sequential decision-making [8], [9], [11]–[13]. Recently, RL-related methods, such as adaptive dynamic programming, have revealed powerful characteristics in optimization and policy decisions [14]–[16]. In RL methods, the agent must discover which action yield the most reward by trying them. Trial-and-error search and delayed reward are the two most important distinguishing features of RL [17]. In [18], it uses Bayesian learning methods to predict the room occupancy over time and uses the classic RL method, Q-learning, to learn the control policy. It is worth mentioning that the air conditioning is controlled using bang–bang control method in [18]. In [19], it uses Q-learning method to learn the control policy and the input of the agent includes time and temperature. Unlike the self-designed reward mechanism in [18], the reward in [19] is given by human's senses. Obviously, due to the limitation of the traditional RL method, the input of the agent can only be low-dimensional data and we cannot learn directly from the high-dimensional data.

In recent years, the deep learning (DL) technology has made great progress. DL is a class of machine learning techniques for supervised or unsupervised feature extraction and for pattern analysis and classification [20]. DL has been applied in the field of RL. This combination is called deep RL. This method is useful in problems with high-dimensional state space and brings new methods to air conditioning control. In [25], it employs the deep Q-network (DQN) method and uses the current time, the

Manuscript received October 15, 2019; revised January 24, 2020 and April 20, 2020; accepted May 24, 2020. Date of publication June 17, 2020; date of current version April 27, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 61722312 and Grant 61533017 and in part by the National Key Research and Development Program of China under Grant 2018YFB1702300. *(Corresponding author: Qinglai Wei.)*

Qinglai Wei and Tao Li are with The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, with the University of Chinese Academy of Sciences, Beijing 100049, China, and also with the Qingdao Academy of Intelligent Industries, Qingdao 266109, China (e-mail: qinglai.wei@ia.ac.cn; litao2019@ia.ac.cn).

Derong Liu is with the School of Automation, Guangdong University of Technology, Guangzhou 510006, China (e-mail: derongliu@foxmail.com).

Color versions of one or more of the figures in this article are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2020.3001849

zone temperature, and environment disturbances as inputs of the agent. The agent adjusts the temperature of the room by adjusting the air flow volume of the air conditioning. Although the method of DL is used, [25] still uses low-dimensional data as inputs, which may not be available for high-dimensional image input. Images are fundamentally different from traditional low-dimensional data and traditional data-based control methods cannot use images as inputs. To the best of the authors' knowledge, there is no discussion on the DQN method for the air conditioning control problem through images of human expressions. This motivates our research.

In this article, aiming to eliminate human sleepiness and improve work efficiency, it is the first time that the DQN method is employed to solve air conditioning control problems, which directly uses a high-dimensional image of the human expression as the input. First, the air conditioning system and deep RL methods are introduced. Second, the image processing algorithm is described. Third, the DQN algorithm are designed for air conditioning systems. Finally, simulation results are presented to show that the present method can effectively eliminate sleepiness and improve work efficiency of people.

II. Related Works

In this section, the development of traditional RL, deep RL algorithms, and the application of deep RL in air conditioning control will be introduced.

A. Reinforcement Learning

The working process of a RL agent is described as follows. At each time step t, the agent observes a state s_t from the environment. It chooses an action a from the set of valid actions according to a given policy π, which is a mapping from state space to action space. After the agent executed the action a, the agent will observe the next state s_{t+1} and get a reward r_t from the environment. The agent will constantly interact with the environment until it observes the terminal state. The agent's emphasis on future rewards is discounted by a factor of γ at each time step. The $return$, which reflects cumulative rewards obtained by the agent at time step t, is defined as $R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'}$, where T is the time-step at the terminal state. In order to measure cumulative rewards when agent executes the action a in the state s, the concept of the action value function $Q_\pi(s,a)$ is proposed as

$$Q_\pi(s,a) = E\left[R_t | s_t = s, a_t = a, \pi\right] \tag{1}$$

which is the expected $return$ starting from the state s, executing the action a, and following the policy π. Each policy π has a corresponding state value function $V_\pi(s)$ and action value function $Q_\pi(s,a)$. After enough exploration and learning, the agent will find the optimal policy π^* that can maximize the cumulative reward and the corresponding optimal state value function $V^*(s)$ and optimal action value function $Q^*(s,a)$. Note that the optimal action value function satisfies the Bellman equation

$$Q^*(s,a) = E_{s_{t+1}}[r_t + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) | s, a]. \tag{2}$$

RL algorithms can be divided into three categories, which are value-based method, policy-based method, and actor-critic method, respectively. In value-based methods, the agent will learn the action value function $Q(s,a)$ which represents the expectation of the cumulative reward obtained by making action a under state s, i.e., $Q(s,a) = E(R_t | s_t = s, a_t = a)$. After sufficient learning, $Q(s,a)$ will approach the optimal action value function $Q^*(s,a)$. Then the optimal policy π^* is generated by $Q^*(s,a)$ through certain processing (e.g., ε-greedy). The most famous value-based method is Q-learning [26]. For Q-learning algorithm, the action value function $Q(s,a)$ is updated according to

$$\begin{aligned} Q(s_t,a_t) \leftarrow\ & Q(s_t,a_t) \\ & + \alpha(r_t + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)) \end{aligned} \tag{3}$$

where α is the learning rate.

B. Deep RL

The DeepMind team has used deep RL algorithms to achieve breakthrough results in the field of video games [22] and Go [23], which has aroused widespread excitement. In [22], the agent learns the process of human playing games and only takes the video screen as the input. After sufficient training, the DQN agent can surpass human's highest level in 49 Atari games. In [23], it combines deep RL methods with planning methods and uses the Monte Carlo tree search method to reduce the computational complexity of the search process. The agent won in the game against top player Lee Sedol, marking the problem of Go being overcome.

C. Deep RL for Air Conditioning Control

In [25], the method of deep RL was first applied in air conditioning control. In [25], the temperature set-point of the air conditioning is fixed and the agent adjusts the air flow rate. The state observed by the agent consists of the current time, the zone temperature, and environment disturbances. The goal of the agent is to minimize the total energy consumption while keeping the room temperature within a certain range. Therefore, the reward received by the agent consists of temperature error and energy consumption. However, in the work of [25], the input of the controller is still low-dimensional data.

III. Design of Learning Control Method for Air Conditioning Systems via Human Expressions

In this section, the self-learning control method using the image of human expressions for air conditioning systems will be introduced in detail. First, the action and state spaces for the air conditioning system will be introduced. Then the state preprocessing algorithm and the reward function will be explained. Finally, the DQN method for air conditioning control will be described in detail.

A. Action and State Spaces

There are three types of action the agent can execute, which are raising, keeping, and lowering the temperature set-point of airflow.

Since the main purpose of control is to eliminate people's sleepiness, the agent will get the image of human expressions from the environment that can directly reflect people's drowsiness as the state s. The state s is a three-dimensional (3-D) matrix with size $M \times N \times 3$, where M and N are called spatial dimensions and 3 is called the channel dimension.

The parameters M and N depend on the pixel size of the image, and parameter 3 indicates that the image has three channels of RGB, where RGB is a color model of image in red (R), green (G), and blue (B) light. Assume that the air conditioning system knows the temperature of the airflow that it blows, which is usually true. The temperature of the airflow is also part of the state. Generally speaking, the state that the agent receives from the environment contains two pieces of information, which includes the image of human expressions and the temperature of airflow.

B. State Preprocessing and Reward Function

In this section, the state preprocessing algorithm and the design of the reward function will be described, respectively.

To reduce the amount of calculation, the image of the human expression is necessary to preprocess. First, face detection is a necessary step in order to get people's expressions. Second, the part of eyes in the human expression will be intercepted separately considering the movement of eyes can reflect the degree of sleepiness. Generally speaking, the preprocessing of the human expression is mainly divided into three steps, which are detecting the human face, intercepting the part of eyes, and merging with the temperature of airflow to obtain a whole state.

In the step of human face detection, first, the original state s, which is an $M \times N \times 3$ matrix, will be converted to grayscale s_g, which is an $M \times N \times 1$ matrix. Then, a trained face detector in [28] can detect the rectangular area where the human face is located from the grayscale state s_g, and then return the coordinates of the upper left corner and the lower right corner of the rectangular area, which can be defined as (a, b) and (c, d), respectively.

In the step of intercepting eyes from the original image, we use a trained facial landmark detector in [29] to estimate 68 coordinates of landmarks from the grayscale state s_g, whose input parameters are the coordinates of the rectangular region (a, b) and (c, d). The distribution of all landmarks is shown in Fig. 1. The result of facial landmark detector is a 68×2 matrix L, where each row represents the coordinates of a landmark. In matrix L, the 37th and 46th from 68 landmarks, defined as (r_x, r_y) and (l_x, l_y), represent the rightmost position of the right eye and the leftmost position of left eye, respectively. Subtracting a constant C from the coordinate (r_x, r_y) and adding C on the coordinate (l_x, l_y), we can get coordinates (e_{rx}, e_{ry}) and (e_{lx}, e_{ly}), which represent the upper left and lower right corners of the rectangular area containing eyes, respectively. Then, cutting the part of eyes form the grayscale state s_g, we can get the grayscale part of eyes e_g. Then adjust its size to $m \times n \times 1$, where m and n are the expected dimensions of the preprocessed states.

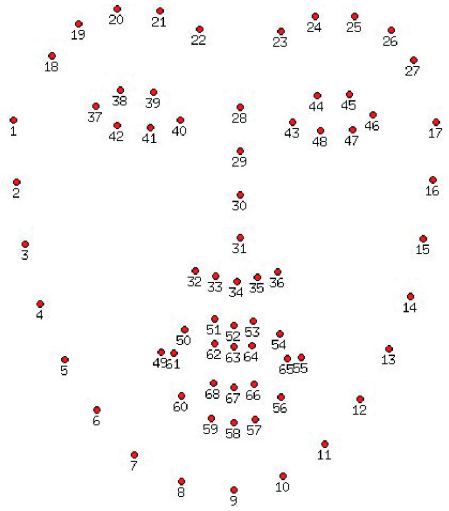


Fig. 1. Sixty-eight facial landmarks. They are used to mark facial features and the outline of the face.

In the step of merging with the temperature of airflow, the temperature is expanded to the same dimensions as the grayscale part of eyes e_g and then the expanded temperature is combined with e_g to be the whole preprocessed state. First, the temperature of airflow t_{air} is expanded to a range of grayscale values 0–255 to get t_{expand} according to

$$t_{\text{expand}} = \frac{t_{\text{air}} - t_{\text{airlow}}}{t_{\text{airup}} - t_{\text{airlow}}} \times 255 \tag{4}$$

where t_{airup} and t_{airlow} represent the highest and lowest temperatures of the airflow blown by the air conditioning. Second, the temperature is expanded to a matrix T_e by

$$T_e = 1_{m \times n} \times t_{\text{expand}} \tag{5}$$

where $1_{m \times n}$ is an $m \times n$ matrix composed of constant 1 and it can be seen that the dimension of T_e is the same as the grayscale part of eyes e_g. Then, the matrix T_e can be extended to the matrix with dimension of $m \times n \times 1$. Third, merge e_g and T_e on the channel dimension to get a whole preprocessed state ϕ which is an $m \times n \times 2$ matrix. The entire preprocessing state algorithm is summarized in Algorithm 1.

The detection of fatigue is critical to the design of the reward function and the degree of fatigue is reflected by the degree of closure of eyes. After detecting the landmarks of eyes, the degree of closure of eyes can be computed easily. In facial landmarks identified by the method in [29], the left and right eyes each occupy six landmarks which are shown in Fig. 2. Inspired by [30], it calculates a value that reflects the degree of eye closure based on coordinates of eyes landmarks called the eye aspect ratio (EAR), which can be computed according to

$$\text{EAR}_r = \frac{||p_2 - p_6|| + ||p_3 - p_5||}{2||p_1 - p_4||} \tag{6}$$

Algorithm 1: Preprocessing of the Original State.

Input:
The state s, which is an $M \times N \times 3$ matrix;
The temperature t_{air} of airflow, which is a single value;
The preprocessed state's dimensions, m and n;
The constant C;
Output:

1: Obtain s_g by converting s to grayscale.
2: Obtain coordinates of the rectangular region (a, b) and (c, d) by using human face detector for s_g.
3: Obtain landmarks' coordinate matrix L by using facial landmark detector for s_g according to (a, b) and (c, d).
4: Obtain the rightmost position of the right eye (r_x, r_y) by extracting L's 37th row.
5: Obtain the leftmost position of the left eye (l_x, l_y) by extracting L's 46th row.
6: $(e_{rx}, e_{ry}) = (r_x - C, r_y - C)$.
7: $(e_{lx}, e_{ly}) = (l_x + C, l_y + C)$.
8: Obtain e_g by intercepting a rectangular area from s_g according to $(e_{rx}, e_{ry}), (e_{lx}, e_{ly})$.
9: $T_e = 1_{m \times n} \times \dfrac{t_{\text{air}} - t_{\text{airlow}}}{t_{\text{airup}} - t_{\text{airlow}}} \times 255$.
10: Obtain ϕ by merging e_g and T_e into an $m \times n \times 2$ matrix.
11: **return**The preprocessed state ϕ.

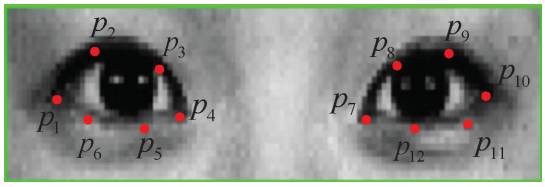


Fig. 2. Twelve landmarks used to mark eyes. The left and right eye each occupies six landmarks which are evenly distributed around the eyes.

$$\text{EAR}_l = \frac{||p_8 - p_{12}|| + ||p_9 - p_{11}||}{2||p_7 - p_{10}||} \tag{7}$$

and

$$\text{EAR}_{\text{eyes}} = \frac{\text{EAR}_l + \text{EAR}_r}{2} \tag{8}$$

where $p_1, p_2, \ldots, p_{12}$ are 2-D facial landmarks. EAR_r and EAR_l represent the EAR of right eye and left eye, respectively. EAR_{eyes} represents the EAR of eyes, which is the average of EAR_l and EAR_r. It can be seen that EAR is equal to the vertical distance of the eye divided by the horizontal distance from (6) and (7).

It has experimentally been verified that when the eye is open, the EAR_{eyes} is approximately constant and when the eye is closed, it quickly drops to nearly 0. In this article, EAR_{eyes} is used to measure the degree of human sleepiness. For a particular person, the maximum EAR (MEAR) is an intrinsic property. This leads to the concept of the relative value of EAR (REAR),

Algorithm 2: Reward Calculation.

Input:
The current s_t, which is an $M \times N \times 3$ matrix;
The last state s_{t-1}, which is also an $M \times N \times 3$ matrix;
Output:

1: Obtain s_{gt} by converting s_t to grayscale.
2: Obtain s_{gt-1} by converting s_{t-1} to grayscale.
3: Obtain L_t and L_{t-1} by processing s_t and s_{t-1} according to steps 1–3 in Algorithm 1.
4: Obtain REAR_t and REAR_{t-1} by computing $\text{REAR}_{\text{eyes}}$ for s_t and s_{t-1} according to (6)–(9) as the input is L_t and L_{t-1}.
5: Obtain the reward r_t at time-step t according to (10) as the input is REAR_t and REAR_{t-1}.
6: **return** r_t.

which is expressed as

$$\text{REAR}_{\text{eyes}} = \frac{\text{EAR}_{\text{eyes}}}{\text{MEAR}_{\text{eyes}}} \tag{9}$$

where $\text{MEAR}_{\text{eyes}}$ represents the maximum aspect ratio of eyes for a particular person and $\text{REAR}_{\text{eyes}}$ represents the relative value of EAR of eyes.

The reward function is important for the agent because it tells the agent what the learning target is. Considering that our goal is to eliminate human sleepiness, the reward function is designed as

$$r_t = \begin{cases} -1 & \text{if } \text{REAR}_t \leq \text{REAR}_{t-1} \\ 1 & \text{if } \text{REAR}_t = 1 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where REAR_{t-1} and REAR_t are $\text{REAR}_{\text{eyes}}$ for $t-1$ and t. The reward r_t is obtained after the agent executed action a_t. If the person's sleepiness is not reduced, which implies that the condition $\text{REAR}_t \leq \text{REAR}_{t-1}$ is satisfied, then the agent is given a punitive reward -1. If the person reaches the state of least sleepiness, which means EAR_{eyes} equals $\text{MEAR}_{\text{eyes}}$, then the condition $\text{REAR}_t = 1$ is satisfied and the incentive reward 1 is given. Otherwise, no reward will be given. The entire process of getting rewards from the environment is summarized in Algorithm 2.

Remark 1: Here, we explain the correct estimating rate of identifying human eyes and compute EAR. The main steps in identifying human eyes and compute EAR are face detection and facial landmark detection. In the step of face detection, we use histogram of gradient (HOG) features and linear support vector machine object detector [28]. In the work of [28], it has tested the algorithm on MIT pedestrian database and this HOG-based detector gives $r_1 = 99.8\%$ accuracy on the MIT test set. In the step of facial landmark detection, we use an ensemble of regression trees (ERT) algorithm that can work in one millisecond [29]. In the work of [29], it has tested the algorithm on HELEN face database and the test error is 0.049. Considering the accuracy requirements of our algorithm for facial landmarks, we consider the results with error less than 0.09 as correct. Let the correct rate of facial landmarks detection

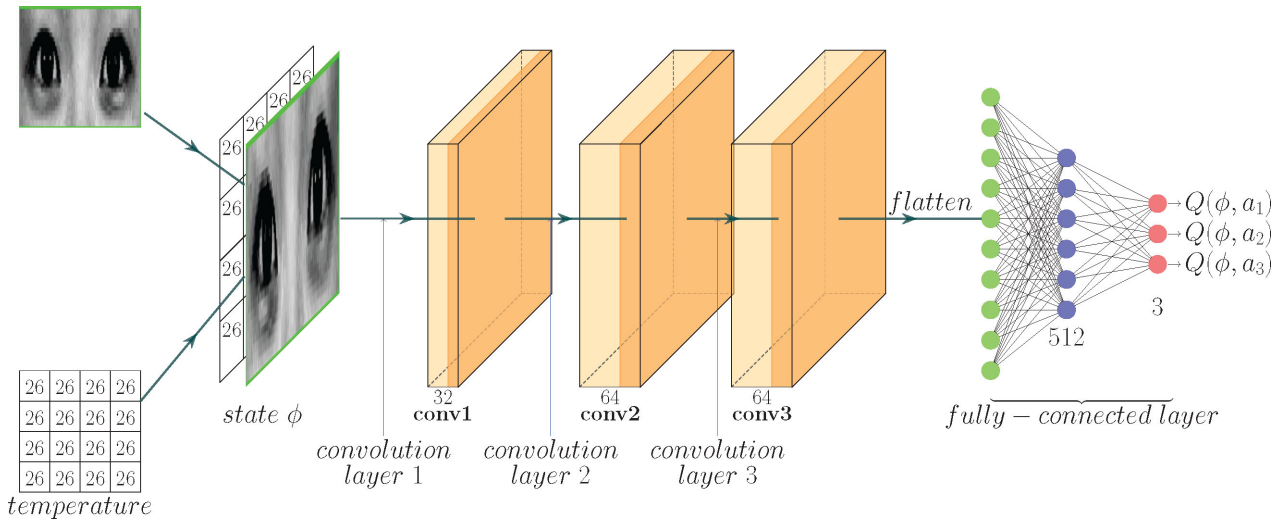


Fig. 3. Architecture of CNN to estimate action value function, which consists of three convolutional layers and two fully connected layers. The input of CNN is the preprocessed state ϕ and the $conv$ represents the output of the convolutional layer.

using ERT algorithm be r_2, then $r_2 = 100\%$. Let the correct estimating rate of identifying human eyes and compute EAR be r. Then $r = r_1 \times r_2 = 99.8\%$. Based on the good test results of face detector and facial landmark detector on the database, we have sufficient reasons to believe that the developed algorithm can achieve good estimating rate in a real office environment.

C. DQN Method for Air Conditioning Control

In this section, the DQN method for air conditioning will be described in detail.

First, the DQN is introduced. DQN method uses a deep neural network to approximate parameterized action value function $Q(\phi, a; \theta)$, which is called Q-network. The Q-network is mainly composed of a convolutional neural network (CNN). Its input is the preprocessed state ϕ in Algorithm 1 and its output is the action value for each valid action. In this article, the CNN consists of three convolutional layers. The input of the CNN is a $50 \times 50 \times 2$ preprocessed state ϕ, where in this article, we set $m = 50$ and $n = 50$. The first convolutional layer is set as 32 filters of 10×10 with stride 2. The second convolutional layer is set to 64 filters of 5×5 with stride 2 and the third convolutional layer contains 64 filters of 3×3 with stride 1. These convolutional layers both use rectified linear unit (ReLU) as the activation function. As the state passes through multiple convolutional layers, the preprocessed state ϕ's spatial dimensions are reduced and the channel dimension is increased. Before going through a fully connected network, the output of final convolutional layer, which is a 3-D matrix, will be flattened into a 1-D vector. The final hidden layer is a fully connected linear layer and consist of 512 ReLU neurons. The output layer is a fully connected linear layer with 3 ReLU neurons, which equals to the number of valid actions. The structure of the entire CNN to estimate the action value function is shown in Fig. 3.

Second, the loss function and the target-network will be introduced. In the DQN method, the action value function $Q(s, a; \theta)$ can be learned by adjusting parameters θ and the target value in the Bellman equation is estimated by $r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta_i^-)$, where $i = 1, 2, \ldots$ is the current number of iterations and θ_i^- is the parameter of previous iteration. The target value is generated by a target-network $\hat{Q}(s, a; \theta_i^-)$ whose structure is identical to the Q-network $Q(s, a; \theta)$. For any $i = 1, 2, \ldots$, the target-network $\hat{Q}(s, a; \theta_i^-)$ will be updated periodically and during the update interval the parameters are fixed. Therefore, for the ith iteration, the loss function is defined as

$$L_i(\theta_i) = \frac{1}{2}\left[r_t + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}; \theta_i^-) - Q(s, a; \theta_i)\right]^2. \tag{11}$$

Third, the experience replay mechanism and training method of the DQN method will be introduced. Inspired by [33], the experience replay mechanism is used in the DQN method. In detail, the agent will store transition (s_t, a_t, r_t, s_{t+1}) at each time-step t into replay memory D. The experience stored in D is always new. In this article, in order to ensure the convergence of the algorithm and save the algorithm's running time and storage space, we repeatedly adjust the parameters by experiments and finally select $N_{\text{init}D} = 500$ and $N_{\text{total}D} = 10\,000$. In this article, the root mean square prop (RMSprop) [32] method is used to train the Q-network. The process of updating the neural network parameters of RMSProp is shown as follows:

$$\boldsymbol{g} = \frac{1}{w}\nabla_\theta \sum_i L\left(Q\left(\phi^{(i)}; \theta\right), y^{(i)}\right) \tag{12}$$

$$\mathbf{r} = \rho \mathbf{r} + (1 - \rho)\boldsymbol{g} \odot \boldsymbol{g} \tag{13}$$

$$\Delta\theta = -\frac{\eta}{\sqrt{\delta + \boldsymbol{r}}} \odot \boldsymbol{g} \tag{14}$$

$$\theta = \theta + \Delta\theta \tag{15}$$

where w is the number of samples, y is the corresponding target of the sample, $\boldsymbol{g}$ is the gradient, $\boldsymbol{r}$ is the cumulative gradient, ρ is

the hyperparameter representing the cumulative gradient decay rate, $\odot$ is element-wise operation, η is the global learning rate, and δ is a small constant to avoid division by zero. In this article, we set the size of minibatch w as 32, the global learning rate η as 0.0025, the decay rate ρ as 0.99, and the small constant δ as 1×10^{-6}.

Finally, the method to increase the agent's exploration of the environment will be introduced. The agent will use the Q-network and ε-greedy method to select the action a_t in the state s_t. It means that the agent has a probability of ε, $0 < \varepsilon < 1$, to select action randomly, with a probability of $1 - \varepsilon$ to choose action $a_t = \max_a Q(s_t, a)$. In order to increase the agent's exploration to the environment, a large ε is used at the beginning of training process, and ε gradually decays as the training progressing. The ε can be computed as

$$\varepsilon = \begin{cases} \dfrac{\varepsilon_{\text{end}} - \varepsilon_{\text{start}}}{t_{\text{decay}}} \times t_{\text{total}} + \varepsilon_{\text{start}}, & t_{\text{total}} < t_{\text{decay}} \\ \varepsilon_{\text{end}}, & t_{\text{total}} \geq t_{\text{decay}} \end{cases} \tag{16}$$

where $\varepsilon_{\text{start}}$ is the initial value of the probability to randomly select action at the beginning of training and ε_{end} is the final value of ε. The parameter t_{total} is the total number of the passed time steps of entire training process. For example, if the training has now proceeded to the tth time step in the ith iteration episode, then t_{total} is equal to the sum of the total number of time steps from the first to ith episode and t. t_{decay} is the total number of the time steps when the probability decays to ε_{end}. Note that $\varepsilon_{\text{start}}$, ε_{end} and t_{decay} are the parameters given before the training starts. In this article, we set the $\varepsilon_{\text{start}}$ as 0.99, the ε_{end} as 0.1, and t_{decay} as 800.

The whole DQN method for air conditioning is described in detail as Algorithm 3.

Remark 2: Compared with the previous methods for air conditioning control methods [2]–[5], [7], the developed method has inherent differences. First, for traditional methods, the input information is low-dimensional and manually selected data. Second, the temperature setpoint was always fixed, and the control purpose is to stabilize the room temperature to the setpoint while saving energy. However, it is pointed out in [7] that control strategies stabilizing the room temperature cannot effectively improve people's work efficiency. Therefore, previous methods are not effective in improving human comfort. In the developed method, the input information is high-dimensional human expression and the features used for decision-making are automatically learned by the algorithm. Furthermore, the control purpose is to control the temperature setpoint so that the room temperature adaptively reaches the most comfortable temperature for a person instead of fixing the room temperature near a given value. That is the advantage of the developed method.

Algorithm 3: DQN Method for Air Conditioning.

Initialization:

Initialize Q-network with random weight θ and target-network with weights $\theta_i^- = \theta$, for $i = 1$;
Initialize state s_t and total time step $t_{\text{total}} = 1$;
Initialize the parameters to compute ε, $\varepsilon_{\text{start}}$, ε_{end} and t_{decay}.
Initialize replay memory D's initially size $N_{\text{init}D}$ and total size $N_{\text{total}D}$;
Initialize the number of episodes trained E, the update interval K and the maximum number of time-steps in an episode T;
Initialize the replay memory D;

Iteration:

Obtain the preprocessed state ϕ_t by preprocessing the state s_t according to Algorithm 1.
for $i \in 1 \ldots E$ **do**
  Initialize state s_t and preprocess it to get ϕ_t.
  for $t \in 1 \ldots T$ **do**
    Compute the ε according to (16).
    With probability ε select a random action a_t, otherwise select $a_t = \arg\max_a Q(\phi_t, a; \theta_i)$.
    Execute action a_t and observe the next state s_{t+1}.
    Get the reward r_t according to Algorithm 2.
    Obtain ϕ_{t+1} by preprocessing s_{t+1}.
    Store $(\phi_t, a_t, r_t, \phi_{t+1})$ in replay memory D.
    Sample a minibatch containing w transitions from D.
    For every transition $(\phi_j, a_j, r_j, \phi_{j+1})$, if ϕ_{j+1} is a terminal preprocessed state then set $y_j = r_j$, otherwise set $y_j = r_j + \gamma \max_{a'} \hat{Q}(\phi_{j+1}, a'; \theta^-)$.
    Train the Q-network using RMSprop.
    $s_t = s_{t+1}$, $\phi_t = \phi_{t+1}$, $t_{\text{total}} = t_{\text{total}} + 1$.
    Every K steps copy parameters of Q-network to target-network.
    Until s_t is a terminal state.
  end for
end for
returnThe Deep Q-network $Q(\phi_t, a_t; \theta_i)$.

IV. Simulation Experiment

In this section, a simulation example is displayed and simulation results are presented to show the effectiveness of the developed DQN method.

A. Simulation Environment Design

In this section, the workflow of the air conditioning system controlled by the facial expression is explained. First, the agent receives the human expression, and then it preprocesses the expression. It uses the Q-network and ε-greedy method to change the temperature setpoint of air conditioning. After that, the temperature of the room will change, which affects the expression of the person. After a period of time, the agent receives the human expression again and readjusts the setpoint temperature of the air conditioning. The schematic diagram of the DQN agent working in the environment is shown in Fig. 4.

In order to simulate the temperature changes in the room, thermal model needs to be established. A simplified room heat

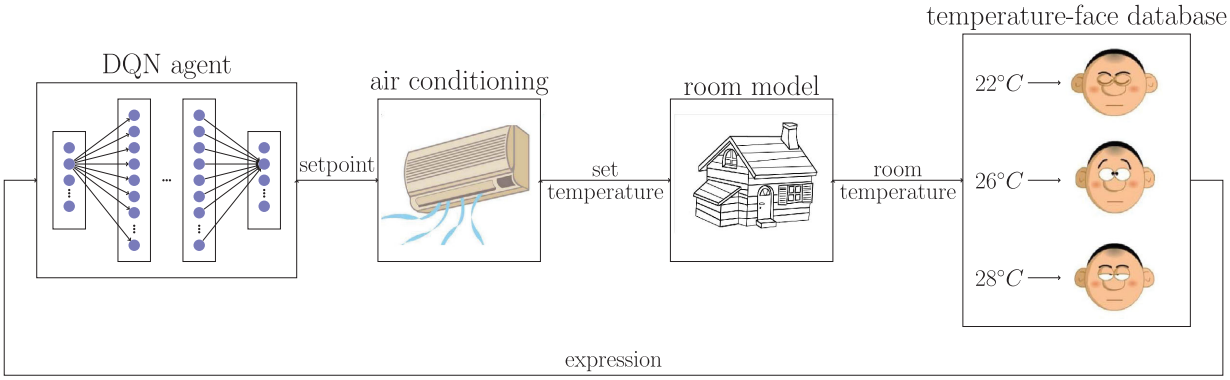


Fig. 4. Architecture of simulation environment.

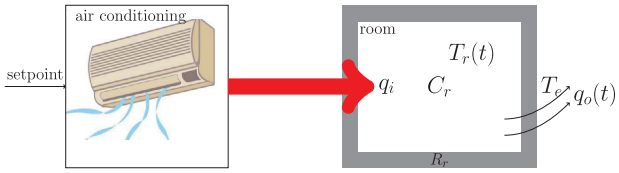


Fig. 5. The simplified room heat model.

model can be described by

$$\dot{T}_r(t) = \frac{1}{C_r}\left[q_i - q_o(t)\right] \tag{17}$$

$$q_o(t) = \frac{1}{R_r}\left[T_r - T_o(t)\right] \tag{18}$$

and

$$\dot{T}_r(t) + \frac{1}{R_r C_r} T_r(t) = \frac{1}{C_r} q_i + \frac{1}{R_r C_r} T_o \tag{19}$$

where T_r and T_o represent the indoor and outdoor temperature, respectively. Parameters C_r and R_r represent the thermal capacitance and thermal resistance, respectively. Parameters q_i and q_o represent input and output heat to the room, respectively. The room model is depicted in Fig. 5. After the temperature setpoint is changed, the heat input to the room changed, and then the temperature of the room changed. According to (17)–(19), the change of room temperature can be described as

$$T_r(t) = R_r q_i + T_o + \left(T_r(0) - R_r q_i - T_o\right) e^{-\frac{1}{R_r C_r} t}. \tag{20}$$

The room temperature will tend to be $t_\infty = R_r q_i + T_o$ over time. Since we assume that the temperature setpoint changes every Δt time (assume that within time Δt, the temperature of room is very close to the steady state value t_∞), the air flow and the temperature setpoint are constant during the time Δt. The heat input to the room q_i is related to the air volume and the air temperature of the air conditioning. During a fixed period of time Δt, the air volume is constant due to a certain air flow rate. So, the q_i is only linear with temperature setpoint. Therefore, the steady-state temperature of room t_∞ is approximately linear with the temperature set point of air conditioning and the parameters of this linear relationship are related to the parameters of the room and the environment. In this article, for the convenience of simulation, we assume that the temperature setpoint of the air conditioning is always 2 °C higher than the steady-state temperature of the room.

TABLE I
TEMPERATURE-FACE DATABASE

Room temperature	EAR	REAR
18	0.096	25.60%
20	0.207	55.56%
22	0.250	67.03%
24	0.313	83.77%
26	0.373	100.00%
28	0.316	84.65%
30	0.264	70.70%
32	0.114	30.53%
34	0.103	27.53%

In order to simulate the change in human expression with temperature, we need a temperature-face database to simulate the performance of people at different temperatures. Its input is the temperature of the room and the output is the image of human face. The design details of temperature-face database are as follows. The people's sleepiness will increase and eyes will close when the temperature is too high or too low because people do not feel comfortable. Eyes are wide open when the temperature is suitable. Therefore, the EAR is the largest when the temperature is suitable, and the EAR becomes smaller when the temperature is too high or too low. In this article, we set the room temperature to take 6 values between 18 °C and 34 °C. In detail, the temperature-face database settings are shown in Table I.As can be seen from Table I, when the temperature is 26 °C, the EAR is the largest and the REAR is 100%.

The specific details of the simulation experiment are described below. At the beginning of each episode, the room temperature will be initialized to a value that is too cold or too hot. In this article, the initial temperature of the room will be randomly selected from 18 °C, 20 °C, 32 °C, and 34 °C. Air conditioning can raise the temperature setpoint, keep it unchanged, and decrease it, respectively. In this article, the air conditioning can only change the temperature setpoint by 2 °C each time. When peoples sleepiness is reduced to a minimum, the iteration episode

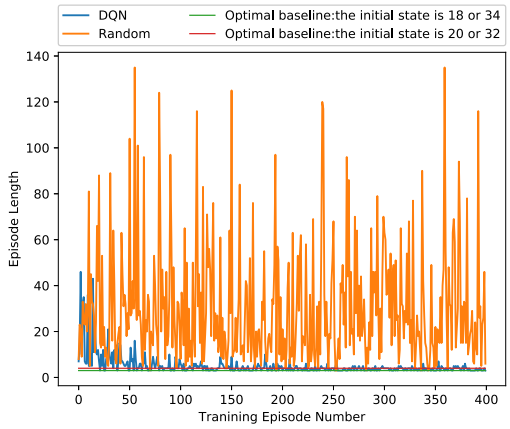


Fig. 6. Episode length gradually decreases during the training process.

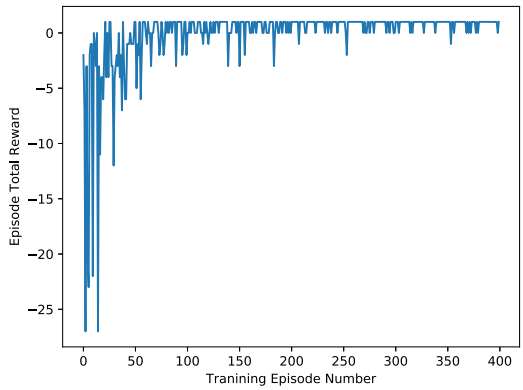


Fig. 7. Episode reward gradually increases during the training process.

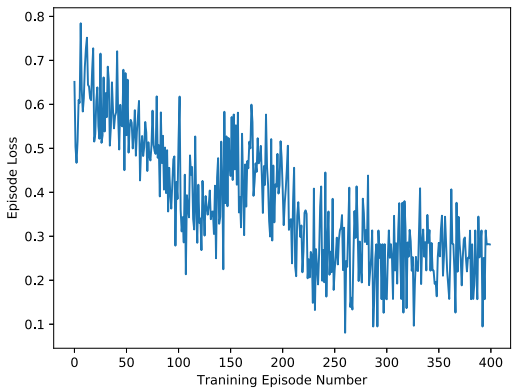


Fig. 8. Episode loss gradually reduces during the training process.

is over and the next will begin. The room temperature will be initialized again.

B. Simulation Results

The episode length is the number of times the agent adjusts the temperature setpoint in each episode and it reflects training results. The episode length during the training process is shown in Fig. 6. The episode length at the beginning of training is significantly larger than the later training period. In the early stages of training, on the one hand, a large ε makes the agent tend to select actions randomly, and on the other hand, an untrained agent cannot make the right decision, which leads to a large episode length. As the training progresses, the agent is gradually able to make the right decision, and ε is also reduced, so the episode length is significantly reduced. When the room temperature is initialized to $18\,^{\circ}$C or $34\,^{\circ}$C, the agent needs to adjust the temperature at least 4 times. When the room temperature is initialized to $20\,^{\circ}$C or $32\,^{\circ}$C, the agent needs to adjust the temperature at least 3 times. Therefore the episode length will be stable at 4 or 3 after enough training. Note that the parameter ε will eventually drop to 0.1 so the choice of agent still has some randomness, which causes the episode length to fluctuate at the end of training.

To show the effectiveness of the developed algorithm, two baseline methods have been added to compared with the developed algorithm. One baseline method called random baseline takes random actions. The other baseline method called optimal baseline can obtain the room temperature and know that the most comfortable temperature of the person is $26\,^{\circ}$. The optimal baseline adjusts the room temperature according to the error between the current room temperature and the target temperature. Therefore, the optimal baseline method takes the least time steps to adjust the room temperature to $26\,^{\circ}$. The results of comparison are shown in Fig. 6. From the comparison with the random baseline method, it is showed that the developed method is significantly smaller in episode length. It can be seen from the comparison with the optimal baseline method that the developed method after training can adjust the room temperature to a suitable temperature with a minimum number of time steps according to the person's expression.

The episode total reward is the sum of rewards obtained by the agent in this episode. The episode reward during the training process is shown in Fig. 7. As the training progresses, the agent can make the right decision in every state which is adjusting the temperature in a direction that eliminates the person's sleepiness, and always get a reward of 0. When it reaches the state where the person is least sleepy, it get the reward of 1. Therefore, the episode total reward is stable at around 1 after sufficient training. Similarly, due to the randomness of agent selection, the curve of the episode total reward fluctuates around 1.

Episode loss refers to the loss of the last time step of the current episode, which reflects the closeness of target value and action value function after this episode. The episode loss during the training process is shown in Fig. 8. In the initial stage of training, the target value generated by target-network $\hat{Q}(s,a;\theta_i^-)$ differs greatly from the action value generated by Q-network $Q(s,a;\theta_i)$, and the loss is large. As the training progresses, Q-network gradually approaches the optimal action value function $Q^*(s,a)$. The target value generated by target-network differs slightly from the action value function $Q(s,a)$, and the loss is stable around 0.

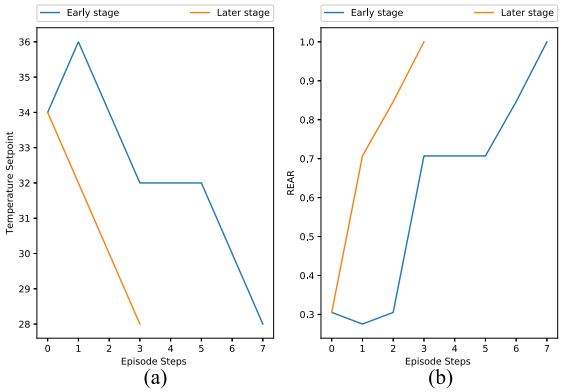


Fig. 9. Temperature setpoint and REAR change trajectories in first and last episode.

In order to show the training results more directly, the change in temperature setpoint and REAR in the first and last episode of the training process is shown in Fig. 9. In the early stage of training, the agent does not know in which direction to adjust the temperature and it takes many time-steps to adjust the room temperature to the most comfortable temperature for the person. People have more time to stay in the state of small REAR. In the later stage of training, the agent can always adjust the room temperature to the most comfortable temperature in the shortest time-steps and people have very little time to stay in the state of small REAR. In the experiment, the most comfortable temperature is $26\,^{\circ}$, but this value is learned by the agent itself rather than preset in advance.

The computation time of the algorithm determines its real-time performance. In this article, the resolution of the image of human expressions is 3456×4608 and the image is colored. The computer for running the algorithm is Acer E5-572G-58HZ and the CPU for running the algorithm is Intel Core i5-4210 M. The programming language used by the algorithm is Python. The construction of the deep neural network uses the TensorFlow package. The state preprocessing algorithm uses the Dlib and OpenCV packages. The developed algorithm is divided into training and decision stages, respectively. In the training stage, the reply buffer is initialized and the Q-network is trained by the back propagation algorithm. Using the hardware mentioned above, the time to initialize the reply buffer is 200.0697 s, or about 3 min, and the time to train Q-network is 1600.458 s, or about 26.5 min. The total computation time of the training stage is 1800.5277 s, or about 30 min. Thus, we say that the training stage is implemented off-line. In the decision stage, the agent receives a state and gives an action signal online. Using the hardware mentioned above, it takes 0.147119 s to make a decision. Considering that temperature is a slowly changing physical quantity, the developed algorithm can be applied in real time at the decision stage. Thus, after training the neural networks, the decision stage is implemented online.

Let the computation time of an episode be t_e. Let the number of times the agent adjusts the temperature in an episode be N_a. Let the running time of the algorithm for a decision be t_w. Using the hardware mentioned above, $t_w = 0.147119$ s. The agent is set to adjust the temperature setpoint every unit time interval Δt. We assume that within time Δt, the room temperature is very close to the steady state. In practical applications, we can set Δt as 15 min. Assume the initial temperature to be $34\,^{\circ}$, then $N_a = 4$. Then, $t_e = (\Delta t + t_w) \times N_a \approx N_a \Delta t = 60$ min.

Remark 3: In this article, the landmark coordinate matrix L is required to implement the Algorithm 1. It is required that the face and eyes can clearly be detected, which implies that the face and eyes are clear without covers and the indoor lighting are bright. Otherwise, it will cause failure of implementing the algorithm. Hence, the disadvantage of the developed algorithm is that the face and eyes are required to be clearly detected without covers and the indoor lighting are required to be bright.

Remark 4: There are several ways to guarantee the control performance when the developed algorithm cannot successfully detect human eyes. First, if lighting factors affect human eyes detection, we can use the latest low-light face detection technology [34] to improve the accuracy. Second, if the covered face caused the failure of eyes detection, we can use a face detection algorithm that is more robust to covered face [35]. Third, if masked eyes make the algorithm unable to obtain eyes information, we can use other available information for fatigue detection tasks. For example, we can use the head posture to detect fatigue [36]. We can also use the previous methods for air conditioning control based on temperature. However, temperature-based methods are not effective in improving human comfort.

V. Conclusion

In this article, the deep RL method was employed to solve air conditioning control problems, which aimed to reduce human sleepiness. The agent directly took the high-dimensional information of the human expression as the input, instead of the temperature data in the previous work in air conditioning control. Through DQN method, the DQN could learn the optimal policy for temperature control directly through the image. Through simulation verification, the DQN agent could learn to adjust indoor temperature through human expressions after sufficient learning.

In the future, we will reduce the computation of the developed algorithm so that it can be used on embedded devices with small computing power and low power consumption. On the other hand, we will expand the algorithm's application to the detection of the face with covers (glasses for example) and dusky indoor lights.

References

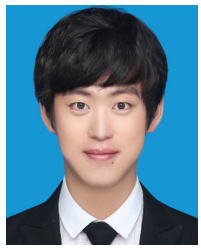
[1] A. Hedge and D. E. Gaygen, "Indoor environment conditions and computer work in an office," *HVAC&R Res.*, vol. 16, no. 2, pp. 123–138, Feb. 2011.
[2] J. Xu and Z. Feng, "Application of two-stage fuzzy PID control in variable refrigerant volume air conditioning systems," in *Proc. Chin. Control Decis. Conf.*, Jun. 2009, pp. 5619–5622.
[3] M. Maasoumy, A. Pinto, and A. Sangiovanni-Vincentelli, "Model-based hierarchical optimal control design for HVAC systems," in *Proc. Dyn. Syst. Control Conf.*, Oct. 2011, pp. 271–278.
[4] S. Yordanova, D. Merazchiev, and L. Jain, "A two-variable fuzzy control design with application to an air-conditioning system," *IEEE Trans. Fuzzy Syst.*, vol. 23, no. 2, pp. 474–481, Apr. 2015.

[5] D. Leducq, J. Guilpart, and G. Trystram, "Non-linear predictive control of a vapour compression cycle," *Int. J. Refrigeration*, vol. 29, no. 5, pp. 761–772, Aug. 2006.
[6] Q. Wei, H. Li, and F.-Y. Wang, "Parallel control for continuous-time linear systems: A case study," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 4, pp. 919–926, Jul. 2020.
[7] W. L. Tse and A. T. P. So, "The importance of human productivity to air-conditioning control in office environments," *HVAC & R Res.*, vol. 13, no. 1, pp. 3–21, 2007.
[8] Q. Wei, L. Wang, Y. Liu, and M. M. Polycarpou, "Optimal elevator group control via deep asynchronous actor-critic learning," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2020.2965208.
[9] Q. Wei, Z. Liao, Z. Yang, B. Li, and D. Liu, "Continuous-time time-varying policy iteration," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2019.2926631.
[10] Q. Wei, Z. Liao, R. Song, P. Zhang, Z. Wang, and J. Xiao, "Self-learning optimal control for ice storage air conditioning systems via data-based adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, to be published, doi: 10.1109/TIE.2020.2978699.
[11] Q. Wei, D. Liu, F. L. Lewis, Y. Liu, and J. Zhang, "Mixed iterative adaptive dynamic programming for optimal battery energy control in smart residential microgrids," *IEEE Trans. Ind. Electron.*, vol. 64, no. 5, pp. 4110–4120, May 2017.
[12] Q. Wei, F. L. Lewis, G. Shi, and R. Song, "Error-tolerant iterative adaptive dynamic programming for optimal renewable home energy scheduling and battery management," *IEEE Trans. Ind. Electron.*, vol. 64, no. 12, pp. 9527–9537, Dec. 2017.
[13] Q. Wei, G. Shi, R. Song, and Y. Liu, "Adaptive dynamic programming-based optimal control scheme for energy storage systems with solar renewable energy," *IEEE Trans. Ind. Electron.*, vol. 64, no. 7, pp. 5468–5478, Jul. 2017.
[14] Q. Wei, R. Song, Z. Liao, B. Li, and F. L. Lewis, "Discrete-time impulsive adaptive dynamic programming," *IEEE Trans. Cybern.*, in press, 2019.
[15] Q. Wei, F. L. Lewis, D. Liu, R. Song, and H. Lin, "Discrete-time local value iteration adaptive dynamic programming: Convergence analysis," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 48, no. 6, pp. 875–891, Jun. 2018.
[16] J. Na, B. Wang, G. Li, S. Zhan, and W. He, "Nonlinear constrained optimal control of wave energy converters with adaptive dynamic programming," *IEEE Trans. Ind. Electron.*, vol. 66, no. 10, pp. 7904–7915, Oct. 2019.
[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[18] E. Barrett and S. Linder, "Autonomous HVAC control, a reinforcement learning approach," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, Porto, Portugal, Aug. 2015, pp. 3–19.
[19] P. Fazenda, K. Veeramachaneni, P. Lima, and U. O'Reilly, "Using reinforcement learning to optimize occupant comfort and energy usage in HVAC systems," *J. Ambient Intell. Smart Environ.*, vol. 6, no. 6, pp. 675–690, Jan. 2014.
[20] L. Deng and D. Yu, "Deep learning: methods and applications," *Found. Trends Signal Process.*, vol. 7, no. 3/4, pp. 197–387, Jun. 2014.
[21] V. Francois-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Found. Trends Mach. Learn.*, vol. 11, no. 3/4, pp. 219–354, Dec. 2018.
[22] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[23] D. Silver *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, Oct. 2017.
[24] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," *arXiv:1312.5602*, Dec. 2013.
[25] T. Wei, Y. Wang, and Q. Zhu, "Deep reinforcement learning for building HVAC control," in *Proc. 54th Annu. Des. Autom. Conf.*, Austin, TX, USA, pp. 1–6, Jun. 2017.
[26] C. J. Watkins and P. Dayan, "*Q*-learning," *Mach. Learn.*, vol. 8, no. 3/4, pp. 279–292, May 1992.
[27] G. Tesauro, "Temporal difference learning and TD-gammon," *Commun. ACM*, vol. 38, no. 3, pp. 58–68, Mar. 1995.
[28] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, San Diego, CA, USA, Jun. 2005, pp. 886–893.
[29] V. Kazemi and J. Sullivan, "One millisecond face alignment with an ensemble of regression trees," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 1867–1874.
[30] T. Soukupová and J. Cech, "Real-time eye blink detection using facial landmarks," in *Proc. 21st Comput. Vis. Winter Workshop*, Rimske Toplice, Slovenia, Feb. 2016, pp. 1–8.
[31] S. Ruder, "An overview of gradient descent optimization algorithms," Jun. 2017, *arXiv:1609.04747*.
[32] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, pp. 26–31, 2012.
[33] L. Lin, "Reinforcement learning for robots using neural networks," Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-93-103, Jan. 1993.
[34] M. Sarin, S. Chandrakar, and R. Patel, "Face and human detection in low light for surveillance purposes," in *Proc. Int. Conf. Comput. Intell. Knowl. Economy*, Dubai, United Arab Emirates, Dec. 2019, pp. 614–620.
[35] S. Ge, J. Li, Q. Ye, and Z. Luo, "Detecting masked faces in the wild with LLE-CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, Jul. 2017, pp. 426–434.
[36] Y. Li, C. Zhang, and L. Meng, "A fatigue driving detection system for train driver based on head pose features," *J. Transport Inf. Saf.*, vol. 32, no. 5, pp. 114–119, 2014.

Qinglai Wei (Member, IEEE) received the B.S. degree in automation, and the Ph.D. degree in control theory and control engineering, from the Northeastern University, Shenyang, China, in 2002 and 2009, respectively.

From 2009–2011, he was a Postdoctoral Fellow with The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China. He is currently a Professor with the institute and the Associate Director of the laboratory. He has authored four books, and published over 80 international journal papers. His research interests include adaptive dynamic programming, neural networks-based control, optimal control, nonlinear systems and their industrial applications.

Dr. Wei is the Associate Editor of several SCI journals. He was the recipient of the Young Researcher Award of Asia Pacific Neural Network Society (APNNS).



Tao Li received the bachelor's degree in automation from Northeastern University, Qinhuangdao, China, in 2019, and is currently pursuing the Ph.D. degree in control theory and control engineering with The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing, China.

His current research interests include adaptive dynamic programming, reinforcement learning, optimal control, and neural network-based control.

Derong Liu (Fellow, IEEE) received the Ph.D. degree in electrical engineering from the University of Notre Dame, Notre Dame, IN, USA, in 1994.

He became a Full Professor of Electrical and Computer Engineering and of Computer Science with the University of Illinois at Chicago, Chicago, IL, USA, in 2006. He has authored or coauthored 19 books.

Dr. Liu is the Editor-in-Chief of *Artificial Intelligence Review* (Springer). He was the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS from 2010 to 2105. He is a Fellow of the International Neural Network Society and International Association of Pattern Recognition.