Journal of Intelligent & Robotic Systems (2020) 97:299–312
https://doi.org/10.1007/s10846-019-01030-0

Efficient Hybrid-Supervised Deep Reinforcement Learning for Person Following Robot

Linzhuo Pang[1] · Yunzhou Zhang[1] · Sonya Coleman[2] · He Cao[3]

Received: 3 May 2018 / Accepted: 23 April 2019 / Published online: 25 May 2019

Abstract

Traditional person following robots usually need hand-crafted features and a well-designed controller to follow the assigned person. Normally it is difficult to be applied in outdoor situations due to variability and complexity of the environment. In this paper, we propose an approach in which an agent is trained by hybrid-supervised deep reinforcement learning (DRL) to perform a person following task in end-to-end manner. The approach enables the robot to learn features autonomously from monocular images and to enhance performance via robot-environment interaction. Experiments show that the proposed approach is adaptive to complex situations with significant illumination variation, object occlusion, target disappearance, pose change, and pedestrian interference. In order to speed up the training process to ensure easy application of DRL to real-world robotic follower controls, we apply an integration method through which the agent receives prior knowledge from a supervised learning (SL) policy network and reinforces its performance with a value-based or policy-based (including actor-critic method) DRL model. We also utilize an efficient data collection approach for supervised learning in the context of person following. Experimental results not only verify the robustness of the proposed DRL-based person following robot system, but also indicate how easily the robot can learn from mistakes and improve performance.

Keywords Person following robot · Deep reinforcement learning · Integration method · Efficient data collection · Real-world situations

1 Introduction

With the development of technologies, the demand for intelligent robots has significantly increased. It would be a great step if robots were capable of moving with their owner as a companion and acting as assistants for people in complex environments such as hospitals, markets or schools. These robots are called person following robots.

However, person following robots face a lot of challenges in real-world environments at present. For example, a robot must be able to distinguish its leader from other confusing objects or similar people and continue its task when the target's shape changes. For outdoor applications, high robustness to illumination is necessary for robots to function properly. Simultaneously, the system should be capable of handling target disappearance and keeping a safe distance from the leader. In addition, obstacle avoidance and path planning are also required when there are obstacles appeared in sight. This paper, however, does not discuss the distance measurement and obstacle avoidance modules. We will focus on the key problems of object occlusion, pedestrian interference, target disappearance and sensitivity to illumination variance in robotic follower application.

Traditionally, the task of person following usually involves two important functions: person tracking and robot navigation. Person tracking involves detecting and tracking a specified person in a crowded environment regardless of surroundings. The output of person tracking algorithm feeds the robot navigation module that is responsible for

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s10846-019-01030-0) contains supplementary material, which is available to authorized users.

✉ Yunzhou Zhang
zhangyunzhou@mail.neu.edu.cn

[1] College of Information Science and Engineering, Northeastern University, Shenyang, China

[2] School of Computing and Intelligent Systems, Ulster University, Coleraine, UK

[3] Faculty of Robot Science and Engineering, Northeastern University, Shenyang, China

generating and sending proper motion control commands when receiving different observations.

Traditional vision-based methods usually need hand-crafted features [1, 2] for the tracking module, and use proportional integral derivative (PID) controllers or fuzzy controllers. At the beginning of a following task, features of the assigned person are extracted and will be used by the detection module to identify positive samples. Once the positions of the target are calculated, control command can be derived by the controller. This design yields high generality to the person following robot system as each time the target person changes, the robot can extract features representing its new owner and begin to follow him/her readily. With some online learning techniques [3, 4], the detection module can be updated when additional information is received. However, the main drawback existing in this kind of person following system is that if the detection module is updated too frequently or the target is interrupted seriously, such as changing his/her pose or disappearing from the view of the camera for a long time, the tracking process will fail, which shows low specificity to the owner.

The proposed approach is to design a person following system to strike a balance between the generality and the specificity. For high specificity, a person following robot should learn to recognize its owner. Features of its owner should be remembered just like humans, so that even the target is interrupted or disappears for some time, the robot can still take proper actions according to what it has learned once its owner appears again. The robot should also be able to improve the knowledge about its owner with more human-robot interaction. For high generality, the robot system should be easy to recognize and follow a new specified person.

The stated goals are achieved in this paper through the utilization of deep reinforcement learning (DRL). Specifically, deep learning enables autonomous feature representation of the assigned person and reinforcement learning enables improvement of the person following policy during interaction. To obtain high generality, we employ hybrid-supervised learning and data expansion techniques, so that the time of policy learning for following a target person is not too long, and we can easily enable the robot system to follow a new specified person. Therefore, the main contributions of this paper are as follows:

1) For real-world person following robot systems, we proposed an end-to-end DRL based technique which can replace manual feature engineering, integrate the tracking and control modules into a complete system and enable the robot to truly recognize its owner.
2) To balance the generality and the specicity in the person following robot system, we proposed a framework. It consists of an efficient data collection approach and an integration method through which an agent receives prior knowledge from a supervised learning (SL) policy network and reinforces its performance with a value-based or policy-based (including actor-critic method) DRL model [5].

2 Related Work

Many researchers are focusing on person following robot methods. Generally, hand-crafted features or convolutional neural networks (CNN) features are used for the tracking module and the PID controllers or fuzzy controllers are used for the controlling module.

When it comes to tracking by hand-crafted features, authors in [2] apply an Optoelectronic (IR) and a fuzzy controller, which enable the robot to track smoothly with high accuracy in any lighting condition. The main drawback is that similar patterns such as chairs and tables may lead to confusion and make it hard to restore from error [6]. A nurse following robot is developed with a high-speed Kinect sensor in [7]. It uses a human skeleton tracking method but is only designed for indoor use due to the limitations of Kinect. Chen et al. [8–11] propose to use stereo cameras to track the person. However, this method is sensitive to light change and needs plenty elaborate sophisticated features to achieve good performance [12]. To overcome these problems, some attempts have been made to fuse data from different sensors. Authors in [13, 14] present fusion of laser range finder (LRF) with infrared camera and separately with an omnidirectional camera. Authors in [15] design a person following robot with a stereo vision-based system and additionally a LRF mounted on the robot body to improve the stability to lighting change. In [16], a human following robot based on a laser range scanner and a Kinect sensor is developed. Other authors have developed a person following robot equipped with a digital camera and RFID system [17]. Owing to the complex data fusion process and inevitable high cost of systems, most research has focused on tracking using only one kind of sensor. In terms of using CNN features for tracking, [18] introduces a stereo vision-based CNN tracker and a PID controller for a person following robot. This is the first attempt to develop a person following robot using deep learning features instead of hand-crafted features. However, the main drawback of this method is that once the target person disappears for a long time, network will be updated with background information, as the online CNN only acts as a feature extractor.

Some shortages still exist in approaches above. Firstly, most of them [6–11] are sensitive to variable illumination and consequently results in low robustness in an outdoor environment. Secondly, target confusion between different

people have not been solved properly in those methods [6–11, 14]. In reality there is a risk that robots may follow another person with similar features. Thirdly, most of the robots will not react properly when the leader is out of view or when the shape of leader changes [6–11, 14]. Fourthly, hand-crafted features have to be carefully designed for robust tracking [6–11, 13, 14]; this greatly increases the manpower cost. Last but not least, the components of human tracking and robot control have to be designed separately to form the complete person following robot system, which means that noise occurred in the tracking module might increase the instability of navigation module.

Deep learning enables feature extraction of much higher robustness compared with traditional hand-crafted features [19, 20]. It allows robot perception systems to learn policies autonomously from raw monocular RGB images, instead of hand-crafted feature detectors [21–24]. Previous research has applied this end-to-end approach into robot systems, among which deep neural networks (DNNs) are the most commonly applied tools [22–26]. For example, in robotic grasping [22], DNNs are used to predict the probability of the motion which will result in successful grasps. However, such successful applications typically rely on large amounts of labelled data and direct supervision of the output, neither of which is available in robotic control. In order to reduce the dependence on real data, recently many researchers try to transfer a deep neural network trained only on simulated RGB images to the real world for the purpose of robotic control such as object localization task [26], object grasping [27] or visuomotor control [28]. Nevertheless, large amount of manual labeling work is still inevitable for these methods.

Emergence of reinforcement learning is considered as a strong artificial intelligence (AI) paradigm that can be used to teach machines through interaction with the environment and learning from mistakes to enhance their abilities [29]. The designers do not need to provide correct labels on a set of situations, all they need to do is to provide a reward signal. A large number of robot control systems have borrowed this idea [30–32]. To accelerate the convergence of the DRL model and decrease robots' training dependence on the real-world environment, some authors propose to pre-train the policy on simulation platforms and extend it to real world [33, 34]. In [35–37], large amounts of random synthetic data are utilized for better generalization when transferring the policy trained in simulation environments to real environments. Nevertheless, some situations in real world are not easy to simulate, which may bring problems when transferring a model from synthetic domains to real-world situations. In [29], the authors take advantage of expert experience to pre-train a model, which can be regarded as providing prior knowledge for the RL agent. Recent work exploits pre-trained visual models to introduce prior information to the perception system [38, 39]. For example, in [39], VGG [40] features are combined with learning predictive dynamics models and reinforcement learning to learn a car following policy on a simulation platform. The main limitation of this method is that the target following performance is restrained by the VGG features trained on the ImageNet dataset [41] that are not optimized for the target following task and their representation may lose some information that is useful for servoing.

In this paper, we formulate the person following task into an action prediction problem, namely predicting which action should be taken on receiving certain observations to follow the assigned person. The policy is trained firstly with supervised learning and then by reinforcement learning. Methods in previous work utilizing DNNs have potential for the person following task. For example, in autonomous driving [25], forest trail following [42] and visuomotor control [23], DNNs are taken as motion classifiers (e.g turn left, turn right and go ahead). However, these methods need a great deal of data annotation to obtain good classification performance, and they do not make use of the interaction between the robots and the environment. Such interactions have potential to boost the robots' performance. For example, when the robot yields an incorrect prediction, feedback from the user may be useful to improve its prediction performance in the future. One solution to this problem is to fine-tune the classification models using online learning [3, 4]. However, the model is prone to being dominated by recent data samples and ignoring historical data. In this paper, we develop an efficient DRL-based framework for a person following robot system, which consists of a hybrid-supervised learning method and an efficient data expansion method for supervised learning in the contexts of person following. With supervised learning and reinforcement learning, the robot can learn not only the pre-trained knowledge, but also knowledge about its owner through the interaction with him/her. Therefore, it can become more intelligent over time. The process of performance enhancing does not require as much labelled data as that of the supervised classification model training. All it needs is a reward signal. With supervised learning and the data expansion approach, the person following robot can autonomously learn the leader's features with minimal data annotation, which helps to acquire general knowledge of its owner in a short time, and the knowledge can be further migrated to DRL models to speed up the convergence of DRL models. Consequently, this framework makes it easier to apply DRL into real-world robotic follower controls and takes account of both the generality and specificity in the person following system.

This work has borrowed the idea in [29], where the policy network is trained by supervised learning, first with expert experience and later using reinforcement learning.

The difference of this paper lies in the fact that the prior knowledge in the SL policy network is migrated to both the value-based and policy-based (including actor-critic method) DRL models to find the one more suitable for the person following task. The connection between value-based and policy-based methods has been established by [5].

This work is also related to [35–37], where they use random synthetic RGB monocular images to train policy in simulation environments that can be generalized to real-world scenarios. For better generalization, a large amount of data has to be produced, which also demands a significant amount of training time. The proposed approach employs few real images from the internet to produce synthetic RGB images to train the person following policy. This is a trade-off between training time and the generalization capability of the model, because images on the internet are not only easy to acquire but also similar to the real scenes. Hence, we can train the policy to be generalizable to the real world with fewer samples.

3 Approach

The problem of person following can be cast into a Markov Decision Process (MDP) [43]. In such a MDP, a state s_t characterizing current observation of robot is composed of N successive frames which are collected by the monocular camera. In each time step t, the robot receives its current state s_t from the camera and then outputs the action $a_t \in \{a_1, a_2, ..., a_M\}$ to specify which direction it should go in order to keep the specified person in the center of the visual field. The mapping from s_t to a_t can be formulated into the policy $\pi_\theta(a_t|s_t) = P[a_t|s_t; \theta]$.

In order to reduce the interaction between the robot and the environment and speed up the convergence of model, this paper applies the integration of supervised learning and unsupervised deep reinforcement learning to the robot. To obtain an agent with satisfactory performance quickly, the RL agent is first provided with heuristic knowledge from a supervised learning (SL) policy network $p_\delta(a|s)$, and then strengthens its performance through interaction with the real world by unsupervised policy-based or value-based DRL models. A synthetic dataset for supervised learning can be collected by making use of a small amount of real web images. This algorithm, consequently, reduces both the volume of labeled data used by supervised learning and the exploration time spent by the DRL agent significantly. Figure 1 shows an overview of the proposed approach to the problem of person following policy learning.

3.1 Efficient Data Collection for Supervised Learning

The most common form of machine learning, deep or not, is supervised learning [44], whose task is to predict the output value for any valid input object after having seen a number of training examples. For a person following robot system, the robot needs to move in the direction of the target person to keep him/her in the center of the sight. For example, when the target person appears in the left side of the field of view, the controller is expected to send a command to turn left. This knowledge can be acquired quickly through supervised learning, where the neural network should be able to predict actions for certain visual input relying on the assigned person's position in an image. The output prediction may be discrete values or continuous values. In this paper, we design the person following robot with discrete action space.

The first thing is to train a network to learn good policy via supervised learning. To obtain enough training sequences, a simple environment makes it possible to extract a specified person from background, which is necessary for labels

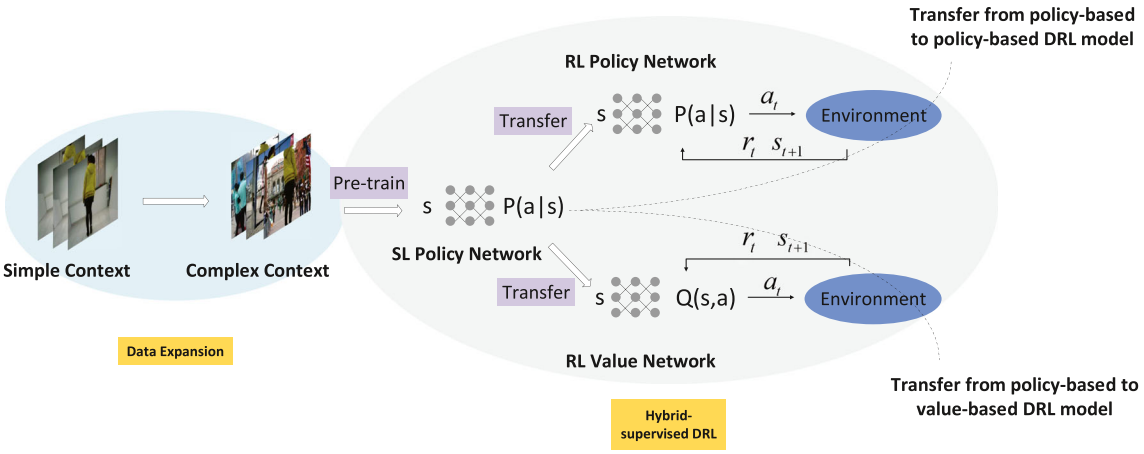

Fig. 1 Whole pipeline of the proposed person following robot system

generation according to the target person's position. In the experiment of this paper, the label set consists of turning right, turning left and going forward. A RL model with random parameters and a larger exploration rate enables the robot to collect samples with target positions of more diversity.

However, this dataset alone is inadequate for the network to learn effective feature representations as the scenes are both simple and homogeneous. Due to the easy extraction of a target person in a simple context, a new data set with complicated situations can be obtained readily through image processing technology (e.g adding the targets to complicated images in this paper), which is able to share the same labels as the original data set. The complicated background pictures can be easily downloaded from the internet. Compared with data expansion methods in [35–37] which produce synthetic data from simulated environments only, the images obtained from the internet represent real situations. Thus, we can produce synthetic data with fewer images and train a model generalizable to the real world with fewer training time. Furthermore, in order to learn illumination invariant features, variations of the V-channel values of images can be introduced to simulate illumination change. This method not only improves data efficiency greatly, but also generates a large amount of data in a short time, containing many complicated scenes so that the neural network learns features of diverse environment with high robustness. This method often appears in DL researches [44]. The simple context we first utilize can be shown in the first row in Fig. 2, where the assigned person can be easily extracted from background.

3.2 Supervised Learning of Policy Network

For the next step of the whole pipeline, we utilize the collected data and build on prior supervised learning knowledge of predicting proper actions in the robotic follower system, so that the robot can obtain general knowledge of its owner in a short time.

Our supervised learning is mainly realized by training a CNN network through the dataset which is efficiently constructed in last section. After training, the CNN network can learn the correspondence between the robot perspective image and the output action. As the network framework we finally use is a reinforcement learning network, this allows us to use the trained CNN network as a policy network for reinforcement learning. The purpose of the policy network is to tell the robot how to take corresponding actions in a certain state.

Generally, the basic idea of the policy network in reinforcement learning is to represent the policy by a parametric probability distribution $\pi_\theta(a|s) = P[a|s;\theta]$ and adjust the policy parameters θ in the direction of greater cumulative reward [45]:

$$\triangle\theta \propto \frac{\partial \log \pi_\theta(a|s)\, Q^\pi(s,a)}{\partial\theta} \tag{1}$$

where $Q^\pi(s,a)$ represents the expected value of doing a in state s and then following the policy π.

The SL policy network $p_\delta(a|s)$ receives the observations of the environment s as input, passes them through many convolutional layers with parameters δ and outputs a

Fig. 2 Dataset preparation pipeline. From left to right: simple context, complex backgrounds, new scenes, illuminance-decreased and illuminance-increased

probability distribution $p_\delta(a|s)$ over possible actions a after the final softmax layer. In terms of the training process, state-action pairs are first randomly sampled from the prepared dataset in which N successive frames (N = 4 in the experiment) captured by the robot represent the states and movements indicated by target person's positions in the frames are actions accordingly. Stochastic gradient ascent is then applied to maximize the likelihood of the proper action a taken in state s:

$$\triangle\delta \propto \frac{\partial \log \pi_\delta(a|s)}{\partial \delta} \tag{2}$$

The structure of the SL policy network is shown in Fig. 3, which takes 4 frames with size $60\times80\times12$ as input and yields a 3-d output, indicating the three actions the robot may take.

3.3 Transferring from Supervised Policy Network to Deep Reinforcement Learning Models

The third stage of the proposed approach aims to make use of the prior knowledge learned by the SL policy network $p_\delta(a|s)$ to accelerate the convergence of DRL models.

The supervised policy network here is the trained CNN network introduced in the previous section. The network provides the prior knowledge of the robot in the following procedure through the training of the dataset. That is, the robot has learned how to take action when meeting some state. Due to the limitations of supervised learning, robots only learn limited knowledge. The purpose of deep reinforcement learning is to make robot learn more knowledge and improve robustness through environmental interaction. Therefore, using the trained CNN network as a policy network for deep reinforcement learning makes the robot continue to interact with the environment with a certain prior knowledge to learn more robust algorithms, which also allows the robot to learn the optimal algorithm faster.

There have been examples of transfer learning from supervised policy learning to reinforcement learning. In [29], the policy network of reinforcement learning takes in the weights of the supervised policy network initially. In [45], a reasonable initial policy is obtained through supervised learning. Their RL optimization with a policy trained with SL substantially accelerates the learning rate of the RL. In [29], the structures and facilities of the two policy networks are identical, which makes it easy to transfer knowledge from one to another. A connection between value-based and policy-based reinforcement learning (RL) has been established in [5], where a strong form of path consistency that relates optimal policy probabilities under entropy regularization to softmax consistent state values for any action sequence has been identified. In this paper, we attempt to apply prior knowledge of the SL policy network to two different DRL models and determine the one with faster convergence in person following policy learning. The two models include: (1) Deep Q-Network by DeepMind (DQN) [46] which is a value-based method and (2) Advantage-Actor-Critic (A2C) [47] which is a policy-based method and outperforms the traditional policy gradient methods [48].

3.3.1 From SL Policy Network to Value-Based DQN

The first attempt is to migrate parameters learned by SL policy network $p_\delta(a|s)$ to value-based DQN $Q_\mu(s,a)$, which has a similar structure of the SL policy network $p_\delta(a|s)$, but removes the final softmax layer to output multiple predictions of state-action values rather than a probability distribution. The weights of DQN are initialized from those of $p_\delta(a|s)$, and are trained by stochastic gradient descent, minimizing the mean squared error (MSE) between the predicted value $Q_\mu(s,a)$ and the corresponding target $r+\gamma \max_{a'} Q_\mu\left(s',a'\right)$, which is the actual current reward plus the discounted estimated future value[46]:

$$\triangle\mu \propto \frac{\partial\left(r+\gamma \max_{a'} Q_\mu\left(s',a'\right)-Q_\mu(s,a)\right)}{\partial\mu} \tag{3}$$

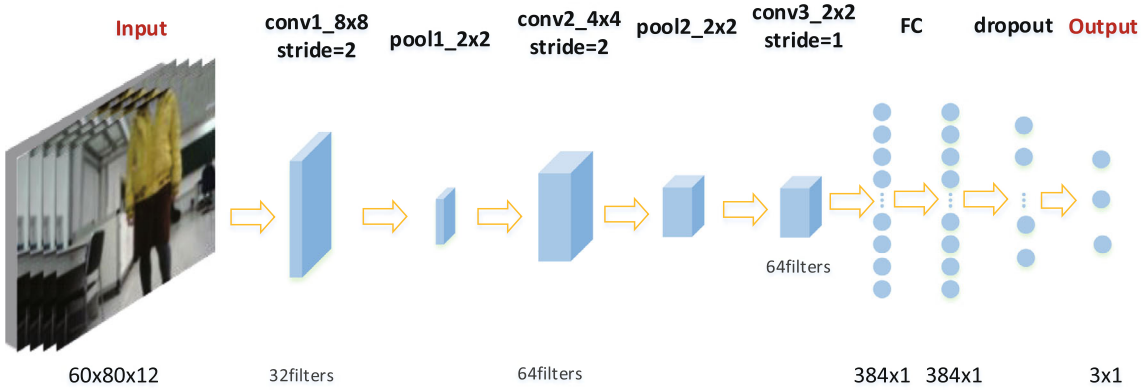

Fig. 3 The structure of SL policy network

3.3.2 From SL Policy Network to Policy-Based A2C

Next, we try to move the prior knowledge of the SL policy network into the famous A2C, which usually utilizes a policy network to act as an actor, and a value network (can be a state-value network $V^{\pi_\theta}(s)$ or an action-value network $Q^{\pi_\theta}(s,a)$ to act as a critic) usually with an advantage function $A(s)$ added to reduce the high variance of policy gradient. The following shows two frequently used frameworks of A2C algorithm:

(1) Estimating both $V^{\pi_\theta}(s)$ and $Q^{\pi_\theta}(s,a)$

One implementation of A2C employs two value networks $V_v(s)$ and $Q_w(s,a)$ to estimate the state-value network $V^{\pi_\theta}(s)$ and action-value network $Q^{\pi_\theta}(s,a)$, both value functions can be updated by, for example, Temporal Difference (TD) learning [49]. Thus the advantage function can be written as:

$$A(s,a) = Q_w(s,a) - V_v(s) \tag{4}$$

and the actor $p_\beta(a|s)$ updates its parameters by :

$$\triangle\beta \propto E[\nabla\beta \log p_\beta(a|s) A(s,a)] \tag{5}$$

(2) Estimating TD-error δ^{π_θ}

Another implementation uses the TD-error δ^{π_θ} to compute the policy gradient, where

$$\delta^{\pi_\theta} = r + \gamma V^{\pi_\theta}(s^{'}) - V^{\pi_\theta}(s) \tag{6}$$

is an unbiased estimate of the advantage function $A^{\pi_\theta}(s,a)$ [33]. Hence, two networks including a policy network $p_\gamma(a|s)$ and a state-value network $V_q(s)$ have been enough to learn the policy. The estimate of TD-error can be obtained in the following way:

$$\delta = r + \gamma V_q(s^{'}) - V_q(s) \tag{7}$$

The parameters updating of $p_\gamma(a|s)$ can be written as:

$$\triangle\gamma \propto E[\nabla\gamma \log p_\gamma(a|s)\delta] \tag{8}$$

However, knowledge from the SL policy network is difficult to transfer to the two frameworks introduced above, because they both contain a state-value network which is significantly different from the SL policy network both in structure and in functionality. The state-value network is to predict the value of certain states while the policy network is to produce a probability distribution of actions. This paper utilizes an A2C framework different from above, attempting to make full use of the pre-trained model.

The structure of A2C in this paper comprises of two networks: a policy network, which shares the same structure as the SL policy network $p_\sigma(a|s)$, and a value network $Q_\tau(s,a)$ which removes the final softmax layer of $p_\delta(a|s)$ and has the same structure as $Q_\mu(s,a)$. This selection of actor and critic can exploit prior knowledge in the SL policy network $p_\delta(a|s)$. Parameters of $p_\sigma(a|s)$ and $Q_\tau(s,a)$ can be both initialized from those of $p_\delta(a|s)$. The objective function and parameters updating for the critic $Q_\tau(s,a)$ is identical to that of DQN:

$$\triangle\tau \propto \frac{\partial\left(r + \gamma \max_{a'} Q_\tau\left(s',a'\right) - Q_\tau(s,a)\right)}{\partial\tau} \tag{9}$$

According to [50], the state-value function for the policy π is:

$$V^\pi(s) = E_\pi\{R_t|s_t = s\} = E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s\right\} \tag{10}$$

and the action-value function for policy π is:

$$\begin{aligned} Q^\pi(s,a) &= E_\pi\{R_t|s_t = s, a_t = a\} \\ &= E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1}|s_t = s, a_t = a\right\} \end{aligned} \tag{11}$$

Therefore, the relationship between the state-value function and the action-value function for policy can be written as:

$$V^\pi(s) = \sum_a \pi(a|s) Q^\pi(s,a) \tag{12}$$

The advantage function in the actor-critic network of this paper can be obtained from $V^\pi(s)$ and $Q^\pi(s,a)$ [46]:

$$\begin{aligned} A_{\tau,\sigma}(s) &= Q_\tau(s,a) - V_\tau(s) \\ &= Q_\tau(s,a) - \sum_a p_\sigma(a|s) Q_\tau(s,a) \end{aligned} \tag{13}$$

The parameters of the actor are then adjusted in a direction of the performance improvement:

$$\triangle\sigma \propto E[\nabla\sigma \log p_\sigma(a|s) A_{\tau,\sigma}(s)] \tag{14}$$

Figure 4 describes how the pre-trained SL policy network is used by the two DRL models. This design can greatly exploit the prior knowledge of the SL policy network, so that the policy learning process of DRL models can be much faster than training a DRL agent from scratch.

4 Experiments and Analysis

In this section, we will introduce the complete experimental system built according to the methods in Section 3 above. The experimental details will be presented in the following order: First, the efficient data set collection for supervised learning. Then, the training of the SL policy network. Next, the specific experiments of the two methods of transferring the supervision policy network to deep reinforcement learning, and the comparison between the

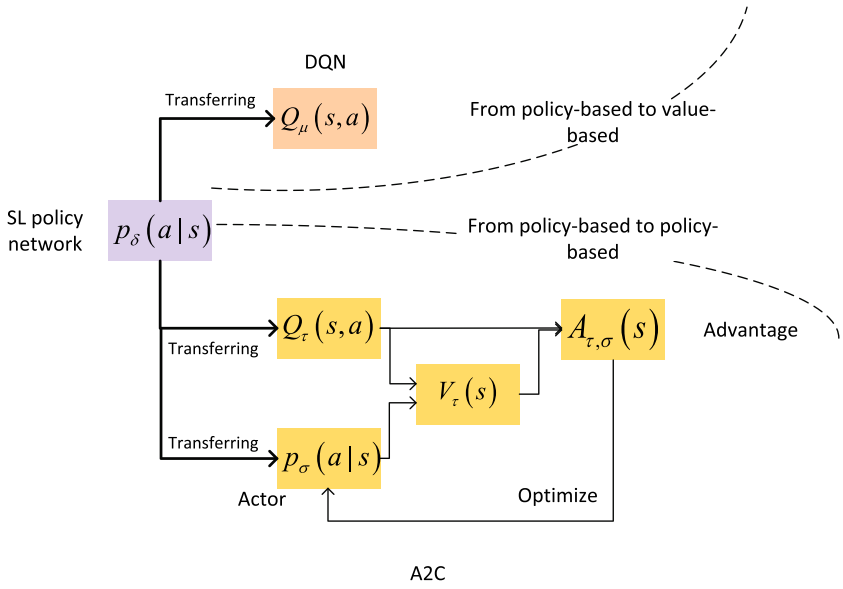

Fig. 4 How the pre-trained SL policy network is used by the two DRL models. From policy-based SL policy network to policy-based A2C model, there are two operations of transferring. From policy-based t value-based DQN, there is one operations of transferring

two weight transfer methods. Finally, we will display the person following robot system built in the physical world and evaluate its performance.

In the experiment, we adopted a TurtleBot2 (only the RGB images are used) robot which is equipped with an Intel NUC6i7KYK with a Quad-core processor. The robotic control is implemented with the help of the Robotic Operating System (ROS). The training process of supervised learning is completed on NVIDIA TitanX. In order to accelerate the learning process of DRL agent, the task of person following is performed locally in the Intel NUC6i7KYK with a processing speed of 12fps, including the computation of action prediction and data collection. The training process of DRL model is performed on the NVIDIA TitanX server using data collected by the local. MySQL Database is utilized to store the data collected during interaction. The two tasks are conducted simultaneously at a specified time interval(5 minutes in this experiment). The latest well-trained model by the server is transferred to the local by Secure Shell (SSH), which will be used by the local at the beginning of the next episode of person following.

4.1 Data Preparations

The main idea of the data expansion approach in this paper is “from simple to complex”. When constructing a simple scene dataset, the number of frames collected by a randomly initialized RL agent in this context is 20, which contain scenes with the different position of the target person at different distance from the robot. In this experiment, we make the target person wear the same clothes in the simple context.

To expand the dataset and enable it to be adaptive to more complicated scenes, we download more than 400 pictures from the internet to simulate the complex backgrounds that the robot may encounter. In order to enable the robot to take proper actions when the target disappears from its view (e.g go forward or circle around, we choose ”circle around” in the experiment), we also add the background pictures to the whole dataset and label them with ”turn right” so that the robot can repeatedly turn right when the leader is out of sight. Thus we can obtain approximately 8400 different frames in total.

By setting certain thresholds of HSV values, the target person can be extracted from the simple context, which will then be added to complex backgrounds. The V value is increased or decreased by a random percentage between $10\% \sim 40\%$ or $10\% \sim 20\%$.

In the 2-bit image of the environment, the pixel values of the person are all 1s and the pixels values of the background are all 0s. Thus person’s horizontal position can be estimated by calculating the average index of pixels with value 1 horizontally. In this paper, we stack 4 successive frames as the input of the networks to infer velocity information. This method was also used in [46]. Therefore, labels can be generated by the average estimated person position, which are all one-hot vectors. Figure 5 shows the relationship between the person position and its corresponding label, where a and b represent the boundaries determining the area of “left”, “middle” and “right”, which are used for the label calculation. Inputs with the same foreground share the same labels. The total time taken for the entire data preparation process is approximately 24 minutes, most of which is spent on generating frames with new complex backgrounds.

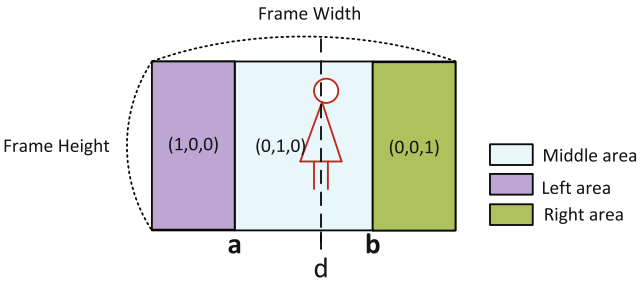

Fig. 5 Relationship between person position and its corresponding label

4.2 Training Details

4.2.1 SL Policy Network Training

The structure of the SL policy network is shown in Fig. 3. Frames collected by the monocular camera in this paper is of size 640×480 pixels. For the real-time nature of the system, input picture needs to be resized to 60×80 pixels to avoid additional forward propagation time added by convolution operations. And it needs to be transformed into HSV format to feed the network.

Since the parameters learned by the SL policy network need to be transferred to DRL models, early stopping technology is utilized during the training process to avoid overfitting and improve generalization ability. We use the Adam optimizer with a learning rate of 1e-6. The training process is stopped after 2000 epochs, which costs almost 20 minutes on four NVIDIA TITAN GPUs.

4.2.2 Transfer Learning of DRL Models

We move the prior knowledge to both the DQN (value-based model) and A2C (policy-based model). By training them with the same data, we determine which one converges faster.

After the weights are initialized, we make the DQN agent interact with the environment. The data collected are then used to train a DQN agent and A2C agent simultaneously. Both the DQN agent and the A2C agent, including the actor and critic, use the Adam optimizer with learning rate of 1e-6. In Fig. 6, the two plots show the average maximum predicted action-value of a held out set of states [49] by DQN and A2C respectively. The number of training samples is approximately 4000. We can see that difference between these two methods is very small when it comes to the convergence speed. Since the robotic follower has discrete action space in this paper, the DQN model is adopted in the application due to its simpler structure. Furthermore, it only needs to migrate the parameters once.

4.3 Person Following Experiment Results

Firstly we test the performance of the robot with the pre-trained SL policy network. Then, after the pre-trained parameters are migrated to the DQN model, we test the easiness and effectiveness of an agent learning from its mistakes. In addition, the proposed DRL-based person following robot is compared with robots using other methods, including KCF-based [51] and TLD-based [52] person following robot, which both use the tracking algorithm to track and the PID controller to control.

4.3.1 Performance of the Pre-Trained SL Policy Network

To confirm the capacity of the pre-trained model, we perform experiments in different environments, including indoor and outdoor contexts, with simple or complicated backgrounds. Results show that the proposed approach can adapt to environments with significant illumination change, object occlusion, shape variance and pedestrian interference. All the images in Figs. 7 and 8 represent the field of vision of the robot. Those images from top to bottom imply the change of the robot's visual field over time.

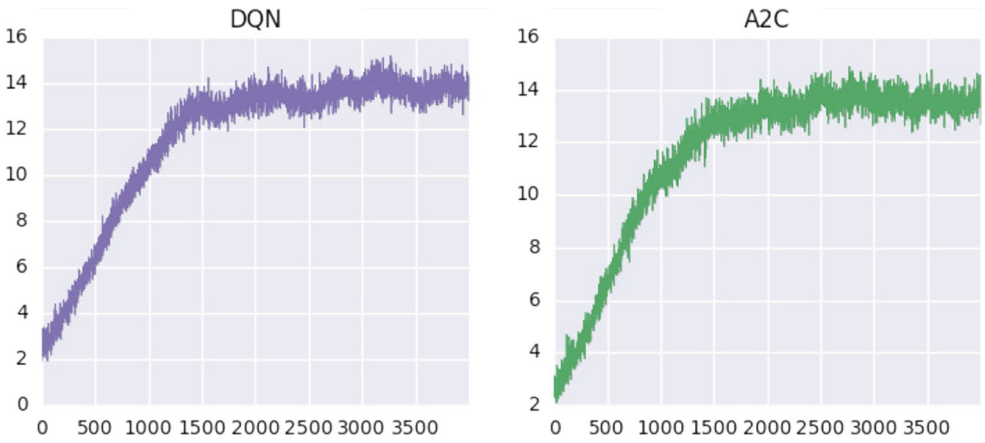

Fig. 6 Average maximum action-value predicted by DQN and A2C. Convergence speed of these two methods is very close

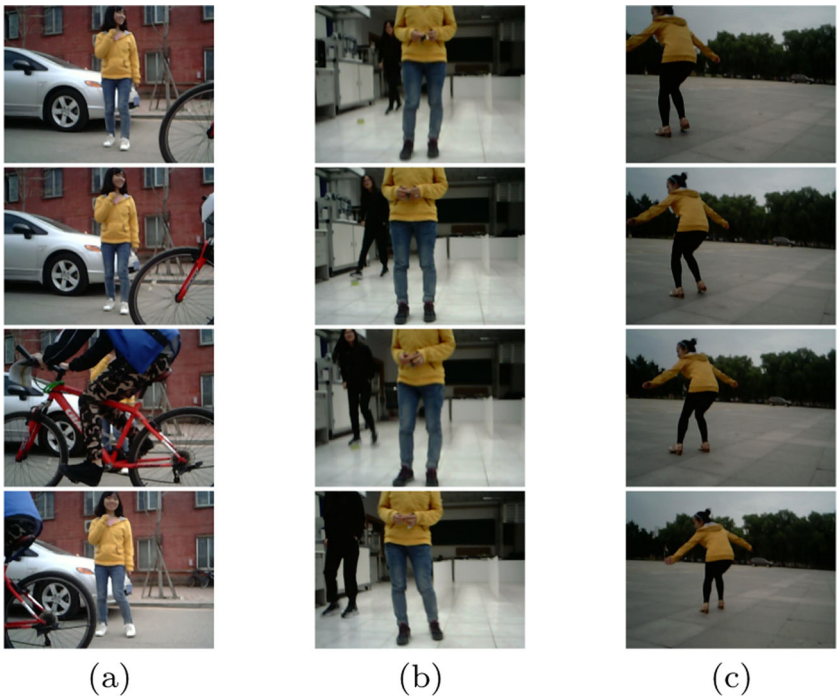

Fig. 7 Experiments with pre-trained model. **a** From top to bottom, robot can always keep its owner in the middle of its view even after object occlusion. **b** From top to bottom, the following task is not influenced by the pedestrian in black. **c** From top to bottom, robot is also robust to shape variance of target person and lower illumination in outdoor environment, and it is always trying to keep the person in the middle of the view

Fig. 8 The DRL agent has corrected its mistake. **a** From top to bottom, the person in yellow is moving away from the middle of robot's view, it seems to mix the person up with doorframe, which has the same color but different shape compared with the target. **b** From top to bottom, it is easy to see that the robot attempts to keep its target in the middle under the disturbance of the yellow doorframe, which means that the agent has learned more about target person's features and corrected the previous mistake

4.3.2 Knowledge Transfer and Reinforcement Learning

While the robot with the pre-trained model runs well the majority of the time, it may be confused with some complex interference, which needs to be overcome by reinforcement learning. In this experiment, we investigate the training epochs and the number of samples required when we want an agent to learn from its mistake. For example in Fig. 8a, the robot seems to mix up the person and doorframe, which have the same color but different shape. When the leader departs from the robot's vision, the DRL agent receives a scalar negative reward and a new episode is started. Data collected during the interaction are fed into DQN to learn a better policy. As shown in Fig. 8b, only approximately 900 training sequences, containing 3 episodes and 100 training epochs which cost approximately 2 minutes, have enabled the agent to correct its previous mistakes and distinguish its owner from the doorframe.

Then, we need to determine the quantity of training samples required for the DRL-based person following robot to adapt to complex outdoor environment. We make the robot follow an assigned person in outdoor environment. During the process, data are collected to train the DRL model repeatedly. The environment used has strong sunlight, cars, walking pedestrians, trees, etc. Experimental results show that the robot can follow its owner for longer periods of time as the experiment continues. After 4 episodes, the robot is able to follow the specified person in the same environment from beginning to end. The target person walks along the preset square path, and the location

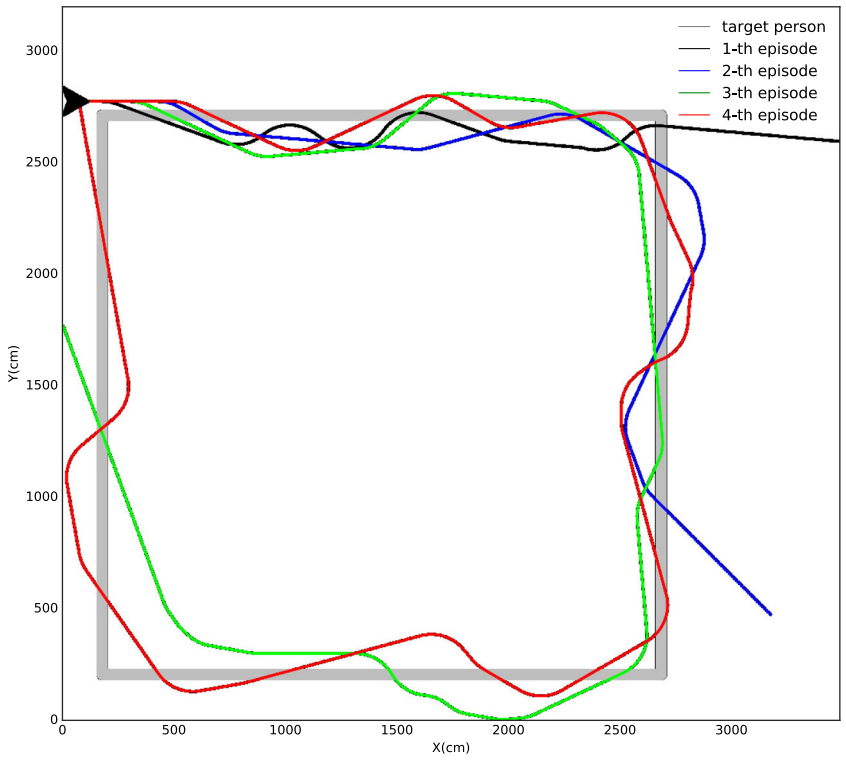

Fig. 9 Paths of robot in multiple experiments. From the black curve to red curve, we can see that at first the robot seems to be attracted by other objects and fails to follow the target person. With more episodes and more training times, the robot can follow its leader for a longer time. After 4 episodes, the robot is able to follow the specified person in the same environment from beginning to end

data of the mobile robot during motion is obtained by the radar positioning and ranging system whose distance measurement error is below 2cm. (The radar system is only used to get the position of the robot.) The path of the robot in each experiment is shown in Fig. 9.

During the experiment, the target person controls the robot with the help of an Android mobile phone, through which he/she can see the visual field of the robot in real time. When the person is almost out of the robot's sight, he/she can stop the episode and start a new one. The agent receives a scalar reward +1 at all states except when the episode is stopped. At that situation, it will receive a negative reward. In this experiment, the negative reward is set to a scalar -10, the following speed of the robot is set to 0.45 m/s, and the turning velocity is set to 15/s.

4.3.3 Comparison Between Person Following Robots Based on Different Methods

To further validate the robustness of the proposed robot following system, we compare the proposed DRL-based robot with person following robots utilizing other existing algorithms.

The first one is the KCF-based person following robot, which extracts the HOG features of a person, tracks him/her using the KCF algorithm [51] and do navigation through a PID controller. The algorithm can be executed very fast. However, the hand-crafted HOG features are very similar among different people, which makes it difficult for the robot to distinguish the leader from others. This experiment is conducted indoors. During the experiment, pedestrian interference is introduced to test robustness. Figure 10a shows the paths of the target person, the DRL-based robotic follower, the KCF-based robotic follower, and pedestrian interference. We can see that the DRL-based robot can follow its leader persistently while the KCF-based one is prone to follow the pedestrian (blue path) during the experiment.

The second approach is the TLD-based [52] robotic follower, which uses a framework designed for long-term tracking of unknown object and a PID controller. This experiment is conducted outdoors with strong sunlight. Both the DRL-based and TLD-based methods are robust to significant light change. However, when the target person disappears (as illustrated with a dot in Fig. 10b), the two methods show different behaviors. The DRL-based robotic follower circles around and continues its following task once the leader is in sight. The TLD-based robot keeps its original motion direction and fails to follow the leader when the leader is in sight again. In order to investigate the behavior of the DRL-based robot, we output the policy of the agent during the period when the target disappears. Results show that all the actions during this period are "right". Since the DRL-based robot has remembered the features of its owner through supervised learning and reinforcement learning, it can continue following the

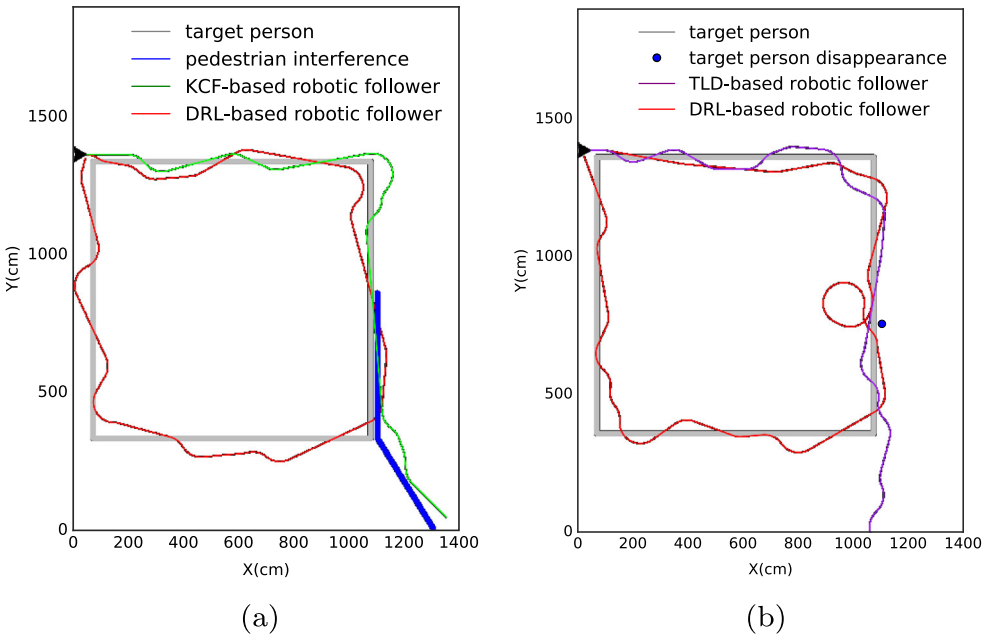

Fig. 10 Comparison between Person Following Robots based on Different Methods. **a** Comparison of the proposed DRL-based robotic follower and KCF-based robotic follower. At first, both the two robots follow well. When the pedestrian appears halfway (blue path), the KCF-based robot begins to follow the pedestrian, while the DRL-based one can follow the owner persistently. **b** Comparison of our DRL-based robotic follower and TLD-based robotic follower. When the person disappears, the former one circles around and continues its following task once the leader is in sight. The latter one is likely to keep its original motion direction and fails to follow the leader when the leader is in sight again

specified person when seeing him/her again. This shows its high specificity to its owner. While for the TLD-based approach, once the leader is out of sight, the background features will come more prominent and change the memory of tracker slowly, which makes it difficult to make correct detection.

The two experiments above are compared with the target tracking algorithm combined with PID controllers. Both the KCF and the TLD are algorithms that achieve good tracking results on the online dataset. As the robot trajectory in Fig. 10, the following effect of these two alternative systems is not good in the physical world.

Our person following robot system is an end-to-end system, that is, from the input image to the execution of the action is a complete whole. Therefore, we can only directly compare with the person following robot which applies target tracking algorithms combined with the PID controller. From the comparative experimental results above, it can be seen that the trajectory of our algorithm can follow the target person robustly. Comparatively, in the two-stage following robot system, the part of the tracking gives a wrong prediction. As a result, an erroneous direction control signal is output after passing through the PID controller, so that the system robustness is lowered. At the same time, we can see from the trajectory of the experimental results that the trajectory of our system always oscillates around the target person trajectory. This is because there are only three kinds of output control actions. In the future, it can be improved by using reinforcement learning with continuous action space.

5 Conclusion

This paper proposed a person following robot system based on deep reinforcement learning. The proposed low-cost system, with a monocular camera, performs well in dynamic environments under challenging situations. To accelerate the whole training process, we introduce an efficient framework which includes the hybrid-supervised reinforcement learning and an efficient data expansion method. The effectiveness is verified through experiments. The framework, consequently, makes it easier for deep reinforcement learning methods to be applied to practical person following applications and make a good balance between the generality and the specificity for the person following robot system. The end-to-end based deep reinforcement learning mechanism not only enables the robot to recognize its owner truly but also enhance its performance over usage time. This yields high specificity to the owner. The hybrid-supervised reinforcement learning and the efficient data expansion enables the robot to acquire general knowledge of the person in a short time, which improves the generality of the system.

6 Limitations and Future Work

During the experiment, we found that once the velocity exceeds 0.6m/s, the person following performance will decrease significantly. There are at least two reasons. One is that the policy computation becomes too slow to catch

up with the robot's action. This can be improved by making use of cloud computing technology. Another reason is that the dimension of output actions is too low to give fine control. This may be solved by continuous robotic control [46] of person following. Furthermore, for the current version, once the target person or his/her clothes change, we need to re-train the model to enable the robot to recognize the new target. This is the main challenge of the proposed method. However, the experimental results show that 50 minutes is sufficient for the robot to recognize a new person and to learn robust following behavior, which is a trade-off between the generality and specificity of the system. Moreover the generality can be improved by introducing one-shot learning [53] and pedestrian identification technology into the system. Future work utilizing more robust features of the person, such as the gait, may also improve the specificity of the person following policy. In addition, distance detection from monocular camera will also be introduced in the future to enhance the robustness.

Acknowledgements This work is supported by the Fundamental Research Funds for the Central Universities(N172608005, N182608004), Young and Middle-aged Innovative Talent Plan of Shenyang(RC170490), Natural Science Foundation of Liaoning (No.20180520040) and National Natural Science Foundation of China (No. 61471110, 61733003).

References

1. Shaker, S.: Fuzzy inference-based person-following robot. International Journal of Systems Applications Engineering & Development (2008)
2. Rashid, M.M., Shafie, A.A., Alamgir, T.B., Alfar, I.J.: Design and implementation of fuzzy based person following mobile robot. In: Applied mechanics and materials, vol. 151, pp. 184–188. Trans Tech Publ (2012)
3. Xiang, Y., Alahi, A., Savarese, S.: Learning to track: Online multi-object tracking by decision making. In: 2015 IEEE International Conference on Computer Vision (ICCV), number EPFL-CONF-230283, pp. 4705–4713. IEEE (2015)
4. Danelljan, M., Bhat, G., Shahbaz Khan, F., Felsberg, M.: Eco: Efficient convolution operators for tracking. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 21–26, Honolulu (2017)
5. Nachum, O., Norouzi, M., Xu, K., Schuurmans, D.: Bridging the gap between value and policy based reinforcement learning (2017)
6. Doisy, G., Jevtic, A., Lucet, E., Edan, Y.: Adaptive person-following algorithm based on depth images and mapping. IEEE. In: RSJ International Conference on Intelligent Robots and Systems (IROS) (2012)
7. Ilias, B., Abdul Shukor, S.A., Yaacob, S., Adom, A.H., Mohd Razali, M.H.: A nurse following robot with high speed kinect sensor. ARPN J. Eng. Appl. Sci. **9**(12), 2454–2459 (2014)
8. Chen, Z., Birchfield, S.T.: Person following with a mobile robot using binocular feature-based tracking. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007, pp. 815–820. IEEE (2007)
9. Satake, J., Miura, J.: Robust stereo-based person detection and tracking for a person following robot. In: ICRA Workshop on People Detection and Tracking, pp. 1–10 (2009)
10. Petrović, E., Leu, A., Ristić-Durrant, D., Nikolić, V.: Stereo vision-based human tracking for robotic follower. Int. J. Adv. Robot. Syst. **10**(5), 230 (2013)
11. Chen, B.X., Sahdev, R., Tsotsos, J.K.: Person following robot using selected online ada-boosting with stereo camera. In: 2017 14th Conference on Computer and Robot Vision (CRV), pp. 48–55 (2017)
12. Atrey, P.K., Anwar Hossain, M., El Saddik, A., Kankanhalli, M.S.: Multimodal fusion for multimedia analysis: A survey. Multimed. Syst. **16**(6), 345–379 (2010)
13. Motai, Y., Jha, S.K., Kruse, D.: Human tracking from a mobile agent: Optical flow and Kalman filter arbitration. Signal Process. Image Commun. **27**(1), 83–95 (2012)
14. Kobilarov, M., Sukhatme, G., Hyams, J., Batavia, P.: People tracking and following with mobile robot using an omnidirectional camera and a laser. In: IEEE International Conference on Robotics and Automation, pp. 557–562 (2006)
15. Tabe, Y., Uesugi, A., Misawa, N., Oguri, T., Omote, E., Igari, J.: Person following robot with vision-based and sensor fusion tracking algorithm. InTech (2008)
16. Hoshino, F., Morioka, K.: Human following robot based on control of particle distribution with integrated range sensors. In: IEEE/sice International Symposium on System Integration, pp. 212–217 (2011)
17. Germa, T., Ouadah, N., Cadenat, V., Devy, M.: Vision and RFID-based person tracking in crowds from a mobile robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5591–5596 (2009)
18. Chen, B.X., Sahdev, R., Tsotsos, J.K.: Integrating stereo vision with a CNN tracker for a person-following robot. In: International Conference on Computer Vision Systems, pp. 300–313 (2017)
19. Antipov, G., Berrani, S.A., Ruchaud, N., Dugelay, J.L.: Learned vs. hand-crafted features for pedestrian gender recognition, pp. 1263–1266 (2015)
20. Budnik, M., Gutierrez-Gomez, E.L., Safadi, B., Quénot, G.: Learned features versus engineered features for semantic video indexing. In: International Workshop on Content-Based Multimedia Indexing, pp. 1–6 (2015)
21. Pomerleau, D.A.: Alvinn: An autonomous land vehicle in a neural network. In: Advances in Neural Information Processing Systems, pp. 305–313 (1989)
22. Levine, S., Pastor, P., Krizhevsky, A., Quillen, D.: Learning hand-eye coordination for robotic grasping with large-scale data collection. International Journal of Robotics Research, (10) (2016)
23. Finn, C., Tan, X.Y., Duan, Y., Darrell, T., Levine, S., Abbeel, P.: Deep spatial autoencoders for visuomotor learning (2015)
24. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J.: End to end learning for self-driving cars (2016)
25. Levine, S., Finn, C., Darrell, T., Abbeel, P.: End-to-end training of deep visuomotor policies. J. Mach. Learn. Res. **17**(1), 1334–1373 (2016)
26. Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 23–30 (2017)
27. Bousmalis, K., Irpan, A., Wohlhart, P., Bai, Y., Kelcey, M., Kalakrishnan, M., Downs, L., Ibarz, J., Pastor, P., Konolige, K.: Using simulation and domain adaptation to improve efficiency of deep robotic grasping (2017)
28. James, S., Davison, A.J., Johns, E.: Transferring end-to-end visuomotor control from simulation to real world for a multi-stage task (2017)

29. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M.: Mastering the game of go with deep neural networks and tree search. Nature **529**(7587), 484 (2016)
30. Vitelli, M., Nayebi, A.: Carma: A deep reinforcement learning approach to autonomous driving (2016)
31. Koutník, J., Schmidhuber, J., Gomez, F.: Evolving deep unsupervised convolutional networks for vision-based reinforcement learning. ACM (2014)
32. El Sallab, A., Abdou, M., Perot, E., Yogamani, S.: End-to-end deep reinforcement learning for lane keeping assist (2016)
33. Pan, X., You, Y., Wang, Z., Lu, C.: Virtual to real reinforcement learning for autonomous driving (2017)
34. Rusu, A.A., Vecerik, M., Rothörl, T., Heess, N., Pascanu, R., Hadsell, R.: Sim-to-real robot learning from pixels with progressive nets (2016)
35. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games, pp. 102–118 (2016)
36. Sadeghi, F., Levine, S.: Cad2rl: Real single-image flight without a single real image (2016)
37. Peng, X.B., Andrychowicz, M., Zaremba, W., Abbeel, P.: Sim-to-real transfer of robotic control with dynamics randomization (2017)
38. Devin, C., Abbeel, P., Darrell, T., Levine, S.: Deep object-centric representations for generalizable robot learning (2017)
39. Lee, A.X., Levine, S., Abbeel, P.: Learning visual servoing with deep features and fitted q-iteration (2017)
40. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computer Science (2014)
41. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Li, F.F.: Imagenet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR 2009, pp. 248–255 (2009)
42. Giusti, A., Guzzi, J., Cireşan Dan, C., He, F.L., Rodríguez, J.P., Fontana, F., Faessler, M., Forster, C., Schmidhuber, J., Di Caro, G.: A machine learning approach to visual perception of forest trails for mobile robots. IEEE Robot. Autom. Lett. **1**(2), 661–667 (2017)
43. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction, vol. 1. MIT Press, Cambridge (1998)
44. Chen, Z., Jacobson, A., Sünderhauf, N., Upcroft, B., Liu, L., Shen, C., Reid, I., Milford, M.: Deep learning features at scale for visual place recognition. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 3223–3230, 05 (2017)
45. Williams, J.D., Zweig, G.: End-to-end lstm-based dialog control optimized with supervised and reinforcement learning (2016)
46. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., Riedmiller, M., Fidjeland, A.K., Ostrovski, G.: Human-level control through deep reinforcement learning. Nature **518**(7540), 529 (2015)
47. Sutton, R.: Policy gradient methods for reinforcement learning with function approximation, vol. 12 (1999)
48. Williams, R.J.: Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach. Learn. **8**(3-4), 229–256 (1992)
49. Silver, D.: Google DeepMind. Tutorial: Deep reinforcement learning (2016)
50. Martin, M.: Reinforcement learning framework (2011)
51. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2015)
52. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1409 (2012)
53. Li, F.F., Fergus, R., Perona, P.: A Bayesian approach to unsupervised one-shot learning of object categories. In: IEEE International Conference on Computer Vision, 2003. Proceedings, vol. 2, pp. 1134–1141 (2008)

Linzhuo Pang received B.S. degree in Automation from Northeastern Univer- sity, Shenyang, China in 2016. She is currently a postgraduate student with the College of Information Science and Engineering, Northeastern University, China. Her research interests include intelligent robot and reinforcement learning.

Yunzhou Zhang received B.S. degree and M.S. degree in Mechanical and Elec- tronic engineering from National University of Defense Technology, Changsha, China in 1997 and 2000, respectively. He received Ph.D. degree in pattern recognition and intelligent system from Northeastern University, Shenyang, China, in 2009, where he is currently a professor with the College of Information Science and Engineering, Northeastern University, China. His research interests include intelligent robot and image processing.

Sonya Coleman received a BSc (Hons.) in Mathematics, Statistics and Computing from the University of Ulster, UK in 1999, and a PhD in Mathematics from the University of Ulster, UK in 2003. She is a Professor of Vision Systems in the School of Computing and Intelligent System at the Ulster University. Prof Coleman has 130+ publications primarily in robotics, image processing, bio-inspired systems and computational finance. Funding from various sources such as EPSRC (EP/C006283/1), The Nuffield Foundation, The Leverhulme Trust and the EU has supported her research. She is currently PI of the Capital Markets Engineering Studentship programme and Co-I on the Capital Markets Collaborative Network project. She is currently Co-I on the EU FP7 SLANDAIL project and was Co-I on recently completed EU FP7 projects VISUALISE and RUBICON. In 2009 she was awarded the Distinguished Research Fellowship by the Ulster University in recognition of her contribution to research.

He Cao received B.S. degree in Automation from Northeastern University, Shenyang, China in 2017. He is currently a postgraduate student with the Faculty of Robot Science and Engineering, Northeastern University, China. His research interests include intelligent robot and image processing.