Robot-Assisted Pedestrian Regulation Based on Deep Reinforcement Learning

Zhiqiang Wan, *Student Member, IEEE*, Chao Jiang, *Student Member, IEEE*,
Muhammad Fahad, *Student Member, IEEE*, Zhen Ni, *Member, IEEE*,
Yi Guo, *Senior Member, IEEE*, and Haibo He, *Fellow, IEEE*

***Abstract*—Pedestrian regulation can prevent crowd accidents and improve crowd safety in densely populated areas. Recent studies use mobile robots to regulate pedestrian flows for desired collective motion through the effect of passive human–robot interaction (HRI). This paper formulates a robot motion planning problem for the optimization of two merging pedestrian flows moving through a bottleneck exit. To address the challenge of feature representation of complex human motion dynamics under the effect of HRI, we propose using a deep neural network to model the mapping from the image input of pedestrian environments to the output of robot motion decisions. The robot motion planner is trained end-to-end using a deep reinforcement learning algorithm, which avoids hand-crafted feature detection and extraction, thus improving the learning capability for complex dynamic problems. Our proposed approach is validated in simulated experiments, and its performance is evaluated. The results demonstrate that the robot is able to find optimal motion decisions that maximize the pedestrian outflow in different flow conditions, and the pedestrian-accumulated outflow increases significantly compared to cases without robot regulation and with random robot motion.**

***Index Terms*—Deep reinforcement learning (DRL), human–robot interaction (HRI), pedestrian flow regulation.**

I. Introduction

DEVELOPING pedestrian crowd regulation approaches can help to avoid crowd accidents in densely populated areas and for emergency evacuation. Existing work on pedestrian crowd regulation includes studies from the perspective of evacuation planning [1]–[3] and optimal design of facilities to improve pedestrian flows [4]–[6]. It was recently proposed to introduce mobile robots to influence the collective motion of pedestrian crowd through human–robot interaction (HRI) [7]–[10]. Studies have been reported on robot-assisted pedestrian regulation in scenarios, such as crossing pedestrian flows [7] and uni-directional pedestrian flow in an exit corridor [8], [10]. In this paper, we propose a novel learning-based scheme for robot-assisted regulation of pedestrian merging flows.

HRI has received considerable attention with the remarkable advance in socially assistive robotics during the past decade. Notably, extensive studies have focused on modeling HRI for human-aware navigation. A traditional motion planning approach is amended with new considerations of socially normative HRI when robots navigate in the human environment [11]. Earlier work focused on modeling HRI explicitly. For example, an HRI model was reported in [12] to describe humans' walking behaviors in the presence of a mobile robot in a mall environment. The model was used for the robot to plan a congestion-free trajectory which, in turn, improves human walking comfort. In [13], a predictive model of human–robot cooperative collision avoidance is developed. The proposed model enabled the robot to navigate safely and efficiently in dense human crowd environments. The performance of the aforementioned model-based approaches may deteriorate due to model inaccuracy and uncertainties in complex human environments. Therefore, learning-based approaches for HRI have drawn considerable attention [14]–[16].

Inspired by the remarkable success in learning control policy from high-dimensional image observation [17], deep learning methods have been exploited to solve challenging robotic problems in real-world environments, such as object recognition [18]–[20], robot navigation [21]–[23], and robotic grasping [24]–[26]. To name a few works, Oliveira *et al.* [20] reported a deep learning methodology based on the convolutional neural network (CNN) for human body part segmentation. The method was tested on real ground and aerial robots and yielded semantically accurate results. In [24], a two-stage cascaded deep network architecture was used for robotic grasp from RGB-D images, which is able to handle multimodal inputs by applying structured regularization. In [26], end-to-end learning of visuomotor policies using deep CNNs was reported, which directly maps image observations

Manuscript received April 10, 2018; revised August 24, 2018 and October 16, 2018; accepted October 17, 2018. The work of Z. Wan and H. He was supported by the U.S. National Science Foundation under Grant CMMI-1526835. The work of C. Jiang, M. Fahad, and Y. Guo was supported by the U.S. National Science Foundation under Grant CMMI-1527016. The work of Z. Ni was supported by the U.S. National Science Foundation under Grant OIA-1833005. This paper was recommended by Associate Editor D. Zhao. *(Corresponding author: Haibo He.)*

Z. Wan and H. He are with the Department of Electrical, Computer and Biomedical Engineering, University of Rhode Island, Kingston, RI 02881 USA (e-mail: zwan@ele.uri.edu; he@ele.uri.edu).

C. Jiang, M. Fahad, and Y. Guo are with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ 07030 USA (e-mail: cjiang6@stevens.edu; mfahad@stevens.edu; yguo1@stevens.edu).

Z. Ni is with the Department of Electrical Engineering and Computer Science, South Dakota State University, Brookings, SD 57007 USA (e-mail: zhen.ni@sdstate.edu).

This paper has supplementary downloadable materials available at http://ieeexplore.ieee.org, provided by the authors.

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

Digital Object Identifier 10.1109/TCYB.2018.2878977

to the torque control output of the robot's motors. These encouraging results have inspired the studies of learning-based methods for complex robotic problems.

In this paper, we formulate a robot motion planning problem for robot-assisted pedestrian flow optimization. A mobile robot is introduced in two merging pedestrian flows, and the goal of the robot motion planner is to find the best motion decision for the robot to efficiently interact with the pedestrians so that the number of pedestrians going through the exit (referred to as "pedestrian outflow" in this paper) is maximized. Due to the complexity in human collective motion behavior and the effect of HRI, it is difficult to use model-based approaches for such a motion planning problem. Our previous work [10] uses adaptive dynamic programming (ADP) to learn HRI online and plan for robot motion. However, the input to the ADP algorithm is the pedestrian features (e.g., pedestrian positions and velocities) extracted from the image of the environment. It requires an additional tracking system, such as that used in [27] and [28], to extract these features from the images. Since the images have rich information about the pedestrians and HRI, it motivates us to use raw images directly for efficient HRI control.

We propose an end-to-end learning control scheme that maps from raw image observation to robot control actions. The end-to-end model extracts the features of environment states from image observation and outputs robot motion decisions to achieve optimal pedestrian flow regulation. A deep reinforcement learning (DRL)-based approach is developed to solve the robot motion planning problem we formulated. Given the images of the environment as input, the proposed approach learns the optimal robot motion that maximizes the pedestrian outflow. Unlike our previous work on learning-based robot motion control [10], where we used the measured features, such as pedestrian velocities and outflow as the algorithm input, the approach proposed in this paper provides end-to-end motion planning from image data of pedestrian flows, which avoids the burden of performing feature detection and extraction, thus saving processing time and improving online learning efficiency. Simulation results show that the robot finds its best positions for HRI, and the pedestrian outflow increases compared to cases without robot regulation and with random robot motion.

The contribution of this paper is two-fold. First, our formulated robot motion planning problem provides a new method to use HRI to optimize pedestrian flows. Compared to our early work that considers the uni-directional exit corridor environment, the proposed merging flow environment is motivated by a real-world crowd disaster scenario and presents much more challenging dynamics due to the bottleneck effect. Second, the proposed DRL architecture achieves end-to-end robot motion planning from sensor images to robot motion decisions. Compared to traditional learning methods such as ADP, this deep neural-network (DNN) structure avoids hand-picking features and utilizes CNN to learn discriminative features to achieve optimal performances. To the best of our knowledge, this is the first time that DRL is used in the HRI study for robot-assisted pedestrian regulation.

The rest of this paper is organized as follows. Section II provides the background and related work. Section III presents the problem formulation of the robot motion planning for pedestrian flow optimization. Section IV provides the DRL algorithm design for the defined problem. Section V presents the simulation results. We conclude this paper in Section VII.

II. Background and Related Work

In this section, we first review the related works on HRI for human-aware robot navigation and pedestrian flow regulation and then the application of DRL in robotic problems.

A. HRI for Human and Robot Collective Motion

HRI has been studied for robot motion planning in the human environment. Particularly, the effect of HRI has been considered in human-aware robot navigation to respect human comfort, naturalness, and social constraints in robot motion. To mention a few, Luber *et al.* presented learning pedestrians' socially aware motion prototypes from real-world pedestrian motion data in [29], where a hierarchical clustering approach was adopted. The learned motion prototypes that respect a comfort distance were used for the robot to establish a dynamic cost map for generating socially acceptable paths among pedestrians. Kim and Pineau [30] proposed a Bayesian inverse reinforcement learning (RL) approach to learn human motion behaviors, which are cast as a cost function in consideration of social variables. The learned cost function, and the observation from the RGB-D sensor, is fed into the robot's path planner to generate socially adaptive motion. The aforementioned works aimed to utilize HRI in a socially adaptive manner, which improves robot acceptance when navigating in the human environment.

On the other hand, it has been found that human motion behavior can be implicitly influenced by passive HRI [7], [9], [10], [31]; namely, the robot moves in a planned motion and the humans adjust their motion to avoid colliding with the robot. In this manner, human collective motion can be modified with a moving robot that dynamically interacts with humans. The effect of passive HRI has been utilized for pedestrian regulation, where desired pedestrian collective motion was achieved with optimized robot motion. The work presented in [9] studied the effect of deploying mobile robots to affect crowd dynamics and showed that the flow efficiency was improved with appropriately designed robot maneuvers and formation patterns. The work shed light on how to utilize passive HRI for controlling and optimizing pedestrian flow. Yamamoto and Okada analyzed the characteristics of passive HRI in a crossing pedestrian flows scenario in [7], and the results were used to find optimal robot motion that can reduce congestion in crossing flows. In our earlier work [10], we proposed regulating the average velocity of a uni-directional pedestrian flow with a robot moving perpendicularly to the flow direction. Due to the effect of passive HRI, the deployed robot behaves as a "virtual gate" that slows down the flow velocity around the robot, and the average flow velocity can be regulated to a desired value by adjusting the robot velocity.

We designed a robot motion control algorithm based on ADP to provide adjustable robot motion online therein.

In this paper, we adopt a pedestrian regulation approach based on passive HRI. We aim to find the optimal robot motion planner that maneuvers the robot to the best positions in which maximum pedestrian outflow can be achieved.

B. Deep Reinforcement Learning in Robotics

In the past two decades, robot planning and control with RL have attracted considerable attention from researchers [32]–[38]. Beom and Cho [32] studied robot navigation in uncertain environments by utilizing RL and fuzzy logic. Asada *et al.* [33] applied an RL framework to train a soccer robot with a look-up table. In these applications, the control policy was learned without the model of the environment. However, the proposed learning algorithms deal with discretized state and action space and, thus, are not applicable to problems with the continuous or high-dimensional state and action space. In order to overcome this drawback, neural-network techniques were developed to handle continuous or high-dimensional input in robotics RL. For instance, Gaskett *et al.* [34] exploited a three-layer perceptron to learn the wandering and visual servoing under continuous state and action space. In [35], a fuzzy neural network enabled the robot to learn basic navigation under a changing environment. However, these approaches used shallow neural networks that have a limited capability of representation learning [36], thus operating on hand-engineered features which require in-depth domain knowledge.

Recently, DNN obtained great success in many complex applications, such as image classification [39]–[42] and object detection [43], [44]. Owing to the strength of DNN, Mnih *et al.* [17] developed a DRL that combines DNN with RL. The DNN consists of a CNN and a Q-network where the CNN was applied to extract useful features from the high-dimensional input images. Then, the Q-network was used to generate actions based on these features. The DNN was trained using RL. The performance of DRL on the Atari games was comparable to that of a professional human player [17]. This achievement was mostly contributed by the representation learning with DNN that enabled automatic feature extraction and end-to-end RL through a gradient descent.

The bloom of DRL techniques [17], [45]–[47] has inspired the development of new methodologies for robot planning and control. Chen *et al.* [48] reported a robot motion planner for decentralized collision avoidance in the human environment. A DRL algorithm was developed to solve the collision-avoidance problem. The algorithm was fed with manually designed features, including the positions and velocities of the robot and other agents. The representation learning capability of deep learning enables robot controllers to operate directly on raw sensory inputs. Various end-to-end leaning approaches have been successfully applied in mobile robot navigation and localization. Tai *et al.* [49] developed a mapless motion planner that only takes the sparse laser rangefinder measurements and the target position as input. The motion planner was trained through DRL without prior demonstrations and manually designed features. In addition to laser rangefinders, image-based visual observations were used as input to end-to-end motion planners. To name a few, Zhang *et al.* [50] proposed a successor-feature-based DRL approach to conduct the navigation task in a simple maze-like environment by using depth images from a Kinect sensor. In [51], a visual navigation problem was studied, where a robot navigated in an indoor environment using only visual observation without a map. A DNN was developed and trained to model the end-to-end robot's policy that generated robot motion action directly from a given visual input.

Although DRL has achieved encouraging progress in robotics, its applications in social navigation where a robot influences human behaviors have not been extensively reported. Recent studies [52], [53] have used RL for pedestrian simulation and show that RL is capable of learning human behaviors. In this paper, we explore applying DRL to a cutting-edge robotic application, namely, learning HRI for robot-assisted pedestrian regulation. We follow the idea of end-to-end learning for robot motion planning and develop a novel DNN-based method to learn a robot motion planner from raw sensor images for optimal pedestrian regulation.

III. Problem Formulation

In this section, we first present the environment setup and then formulate the robot motion planning problem.

A. Motivation and Environmental Setup

Inspired by the empirical study [54] of a real-world crowd stampede incident that caused significant casualties in Mina/Makkah in 2006, we are interested in pedestrian regulation in an environment where two pedestrian flows merge from different directions going through a bottleneck area. Understanding the complex merging behavior of pedestrian flows is of profound practical importance to avoid such pedestrian crowd incidents; thus, studies have been conducted to find out the underlying causes and seek solutions [54]–[57]. In [54], the transition from laminar flow to turbulent flow that caused the stampede was observed by analyzing the video recordings. The quantity, crowd pressure, was proposed to quantitatively measure the buildup of the pedestrian crowd.

To avoid the buildup of crowd pressure that leads to crowd stampedes, crowd regulation is required. The existing work has reported solutions that either optimize the geometric design or the placement of facilities [4]–[6], [58]. However, the design of the stationary architecture and facilities cannot adapt to the real-time change of pedestrian flows. For example, it has been found in [6] and [58] that the regulation performance of pillar-like obstacles depends on the crowd density, and the optimal placement of such obstacles varies under different levels of crowd density. In this paper, we adopt a pedestrian regulation approach based on passive HRI for the merging human flow scenario, where the robot moves in a planned and controlled motion, and the pedestrians around the robot adapt their motion to avoid potential collisions with the robot. Thus, the motion of the robot affects the collective pedestrian flows through passive HRI. Note that the effect of passive HRI on pedestrian flows was previously validated in

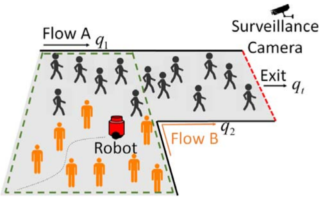


Fig. 1. Merging pedestrian flow scenario. The green-dashed rectangle indicates the robot workspace and the red-dashed line indicates the exit where the instantaneous pedestrian outflow q_t is measured.

simulations [10] and in experiments [59]. Particularly, in [59], the results showing the effect of passive HRI are consistent with simulations using social force models. This paper also shows that the use of a robot is acceptable to humans, similar to a stop sign or a stationary object placed in front of an exit. In such a way, the robot deployed in the environment acts as a moving obstacle that adaptively adjusts its position to regulate the pedestrian flows. As the pedestrian flow condition changes, the robot needs to find the optimal position to prevent the formation of high crowd density and to mitigate congestion, thus maximizing the pedestrian outflow through the bottleneck under different flow conditions.

In this paper, we study the optimal motion planner for the robot to move to the best positions that maximize the pedestrian outflow. Specifically, our environment setup is shown in Fig. 1. Two pedestrian flows A and B merge before the corridor and move toward the exit. The amount of instantaneous inflow A and inflow B are represented by q_1 and q_2, respectively. The entrance of the corridor becomes a bottleneck when the merged flow exceeds the capacity of the exit corridor. Our task is to prevent this congestion using a moving robot through HRI so that the pedestrian outflow from the corridor can be maximized. In Fig. 1, the robot workspace is represented by a green-dashed rectangle, and the outflow is denoted as q_t, which is defined as the number of pedestrians passing through the exit at time step t.

B. Learning Problem Formulation

A finite Markov decision process (MDP) with discrete time step $t = \{1, 2, \dots, T\}$ is used to formulate our robot motion planning problem. MDP provides a mathematical architecture to model a sequential decision-making process. An MDP is defined as the five-tuple $(\boldsymbol{X}, \boldsymbol{U}, \boldsymbol{P}(\cdot,\cdot), q(\cdot,\cdot), \gamma)$, where $\boldsymbol{X}$ represents the observation space of the system, $\boldsymbol{U}$ denotes a set of permissible actions, $\boldsymbol{P}(\cdot,\cdot)$ represents the observation transition model, $q(\cdot,\cdot)$ is the immediate reward, and γ denotes a discount factor. We define our robot motion planning problem using these five elements as follows.

1) Observation Space: The observation $\boldsymbol{x}_t$ is the image of the pedestrians in the environment at time step t. It not only contains the positions of the pedestrians but also shows other features about the pedestrians that are essential for our robot motion planner, such as density.

2) Action Space: In our formulation, the robot workspace is discretized as a regular grid. The robot moves on the grid points. The action space $\boldsymbol{U}$ contains four permissible directions of the robot motion decision $\boldsymbol{u}_t$, that is, "up," "down," "right," and "left," in the grid-based map defined in the robot's workspace.

3) Observation Transition: The transition from observation $\boldsymbol{x}_t$ to the next observation $\boldsymbol{x}_{t+1}$ is defined as

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t). \tag{1}$$

The robot position is updated according to the robot motion decision $\boldsymbol{u}_t$. Then, the robot influences the pedestrian motion through HRI, and the system observation transits from $\boldsymbol{x}_t$ to $\boldsymbol{x}_{t+1}$. It is challenging to model the accurate observation transition function for pedestrians because the HRI and the human motion are exceedingly complex. In this paper, a data-driven approach is proposed to provide control action.

4) Reward: The reward is the instantaneous outflow q_t which is the number of pedestrians passing through the exit at time step t.

In this paper, an action-value function, $Q_{\boldsymbol{\pi}}(\boldsymbol{x}, \boldsymbol{u})$, is defined to assess the performance of taking a motion decision $\boldsymbol{u}$ when given an observation $\boldsymbol{x}$ under a robot motion planning policy $\boldsymbol{\pi}$. This policy maps from each system observation to the probability of taking the robot motion decision. $Q_{\boldsymbol{\pi}}(\boldsymbol{x}, \boldsymbol{u})$ is defined below as the expected sum of rewards starting from observation $\boldsymbol{x}$, taking the motion decision $\boldsymbol{u}$ by following policy $\boldsymbol{\pi}$, that is:

$$Q_{\boldsymbol{\pi}}(\boldsymbol{x}, \boldsymbol{u}) = \mathbb{E}_{\pi}\left[\sum_{k=0}^{K} \gamma^k q_{t+k} \middle| \boldsymbol{x}_t = \boldsymbol{x}, \boldsymbol{u}_t = \boldsymbol{u}\right] \tag{2}$$

where K represents the number of future time steps, $\mathbb{E}_{\boldsymbol{\pi}}[.]$ denotes the expected value given that the robot follows policy $\boldsymbol{\pi}$, and γ balances the importance between the future rewards and the immediate reward. In this paper, γ is set to 1, which means that the future rewards have the same importance as the immediate reward. Since q_t is the instantaneous outflow, $Q_{\boldsymbol{\pi}}(\boldsymbol{x}, \boldsymbol{u})$ is equivalent to the accumulated outflow in the horizon of K time steps.

The objective of the robot motion planning problem is to determine the optimal policy $\boldsymbol{\pi}^{\boldsymbol{*}}$ which maximizes the expected sum of rewards as

$$Q^*(\boldsymbol{x}, \boldsymbol{u}) = \max_{\pi} Q_{\boldsymbol{\pi}}(\boldsymbol{x}, \boldsymbol{u}) \tag{3}$$

where $Q^*(\boldsymbol{x}, \boldsymbol{u})$ represents the optimal action-value function.

Thus, following the optimal policy $\boldsymbol{\pi}^{\boldsymbol{*}}$ and starting from an arbitrary initial observation $\boldsymbol{x}_{\mathbf{1}}$, we can maximize the number of pedestrians passing through the exit in the horizon of K time steps, that is, $Q^*(\boldsymbol{x}_{\mathbf{1}}, \boldsymbol{u}) = \max \sum_{k=0}^{K} q_{k+1}$.

IV. Proposed Approach

Traditional learning-based approaches rely on feature extraction that is designed with experience. This feature engineering process may neglect some important features since finding all useful features requires thorough insight into the problem. These features, however, can be easily observed in the images of the environment, which have rich information on the state of the robot and the pedestrians. Also, it is challenging to find the optimal robot motion policy $\boldsymbol{\pi}^*$ using high-dimensional images. To tackle the challenges, a CNN is applied to extract discriminative features from the input image. The CNN can learn to extract suitable features, based on which a Q network is applied to generate robot motion decisions.

The overall diagram of the DRL-based robot motion planning for pedestrian regulation is illustrated in Fig. 2. A simulated environment is used in this paper. The image of the simulated environment is passed into a DNN to approximate the action value of robot motion decision $\boldsymbol{u}$. The decision $\boldsymbol{u}_i$, with the highest action value, is selected for the robot. It is worth noting that the proposed approach is end to end; that is, it learns the optimal robot motion planning policy directly from the image.

A. Architecture of the Deep Neural Network

As discussed above, the architecture of the DNN shown in Fig. 2 consists of a CNN and a Q network, the details of which are presented next.

1) CNN: A two-layer CNN is implemented to extract discriminative features from the image to facilitate the robot motion planning process. The convolutional layers (Conv 1 and Conv 2 in Fig. 2) have several feature maps which are represented by green and blue squares in the figure. The feature map in the Conv 1 layer is obtained by applying a convolution operation to the input image. If we denote the jth feature map as $\boldsymbol{z^j}$, whose filters are determined by the weight matrix $\boldsymbol{W^j}$ and bias $\boldsymbol{b_j}$, then the feature map $\boldsymbol{z^j}$ is obtained as

$$z^j_{n,v} = g\left(\boldsymbol{b_j} + \sum_{l=1}^{L}\sum_{m=1}^{M} \boldsymbol{W^j_{l,m}} * o_{n+l,v+m}\right) \tag{4}$$

where (l, m) represents the index of the weight matrix whose dimension is $L \times M$, (n, v) denotes the coordinates on the feature map, $g(.)$ denotes the rectified linear unit (ReLU) activation function, and $\boldsymbol{o}$ represents the input image. Similarly, the feature map in the Conv2 layer is obtained by applying the convolution operation to the Conv1 layer. Then, these feature maps are flattened into a vector $\boldsymbol{d}$. More details about CNN can be found in [60].

2) Q Network: The features extracted by the CNN are fed into the Q network, a three-layer fully connected neural network. This type of neural network with a finite number of hidden units can uniformly approximate continuous functions [61]. The value of the hidden unit can be calculated as

$$\boldsymbol{h} = g(\boldsymbol{W_h} * \boldsymbol{d} + \boldsymbol{b_h}) \tag{5}$$

where $\boldsymbol{W_h}$ and $\boldsymbol{b_h}$ represent the weights and the bias, respectively. Then, the hidden layer is connected with the output layer. The value of the output unit is the estimation of the action-value for robot motion decision $\boldsymbol{u}$ when given the input observation $\boldsymbol{x}$, i.e.,

$$Q(\boldsymbol{x}, \boldsymbol{u}) = g(\boldsymbol{W_o} * \boldsymbol{h} + \boldsymbol{b_o}) \tag{6}$$

where $\boldsymbol{W_o}$ and $\boldsymbol{b_o}$ represent the weights and the bias, respectively. Finally, the robot motion decision with the largest action-value is outputted, that is, $\boldsymbol{u} = \text{argmax}_{\boldsymbol{u}\in \boldsymbol{U}}\, Q(\boldsymbol{x}, \boldsymbol{u})$.

Algorithm 1 Training of the DNN

```
Input: Image observation $\boldsymbol{x}$ and reward $q$
Output: DNN's parameters $\theta$
 1: Randomly initialize main DNN's parameters $\theta$.
 2: Initialize a target DNN with parameters $\bar{\theta} = \theta$.
 3: for Epoch=1:N do
 4:     Initialize the environment and obtain observation $\boldsymbol{x}_1$.
 5:     for Time step $t$ = 1:T do
    // apply $\epsilon$-greedy search method
 6:         Sample $c$ from a uniform distribution $\mathcal{U}(0, 1)$.
 7:         if $c > \epsilon$ then
 8:             Obtain $\boldsymbol{u}_t$ from the main DNN.
 9:         else
10:             Randomly select $\boldsymbol{u}_t$ from the action set $\boldsymbol{U}$.
11:         end if
12:         Update robot position $\boldsymbol{p}_t$, observe reward $q_t$,
            and obtain the next image observation $\boldsymbol{x}_{t+1}$.
13:         Store the tuple $(\boldsymbol{x}_t, \boldsymbol{u}_t, q_t, \boldsymbol{x}_{t+1})$ in buffer $\mathcal{B}$.
14:         Randomly sample a batch of tuples
            $\{(\boldsymbol{x}_j, \boldsymbol{u}_j, q_j, \boldsymbol{x}_{j+1})\}_{j=1}^{D}$ from $\mathcal{B}$.
15:         $y_j \longleftarrow q_j + \gamma Q(\boldsymbol{x}_{j+1}, \text{argmax}_{\boldsymbol{u}}\, Q(\boldsymbol{x}_{j+1}, \boldsymbol{u}; \theta_t); \bar{\theta})$.
16:         Calculate the loss function
            $L(\theta_t) = \frac{1}{D}\sum_{j=1}^{D}[y_j - Q(\boldsymbol{x}_j, \boldsymbol{u}_j; \theta_t)]^2$.
17:         Update parameters $\theta_{t+1} = \theta_t - \eta \nabla_{\theta_t} L(\theta_t)$
18:         Every $S$ steps reset $\bar{\theta} = \theta$.
19:     end for
20: end for
```

B. Training of the Deep Neural Network

Algorithm 1 illustrates how to train the DNN according to double Q-learning [62]. The input of Algorithm 1 is the image observation $\boldsymbol{x}$ and the reward q, respectively. Its output is the DNN's parameters θ. Parameters θ include the weights and bias of the CNN and the Q network, that is, $\boldsymbol{W^j}$, $\boldsymbol{b_j}$, $\boldsymbol{W_h}$, $\boldsymbol{b_h}$, $\boldsymbol{W_o}$, and $\boldsymbol{b_o}$.

In Algorithm 1, the DNN in Fig. 2 is referred to as the main DNN. In line 1, the main DNN's parameters θ are randomly initialized. Then, we initialize a target DNN, and its parameters $\bar{\theta}$ are cloned from the parameters of the main DNN. In the outer loop starting from line 3, the parameters θ are iteratively updated through N epochs. At the beginning of each epoch, we first initialize the environment; that is, the pedestrian and the robot positions are initialized. Then, the initial image observation $\boldsymbol{x_1}$ is obtained. In the inner loop starting from line 5, the environment evolves for T time steps. At each time step, from lines 6 to 11, an ϵ-greedy search method [63] is applied to choose the action $\boldsymbol{u}_t$. Specifically, a random number c is generated from a uniform distribution on the interval $(0, 1)$. If c is larger than a constant ϵ, the action $\boldsymbol{u}_t$ is generated by the main DNN in Fig. 2. Otherwise, the action $\boldsymbol{u}_t$ is randomly chosen from the action set $\boldsymbol{U}$. After that, in line 12, the robot position is updated according to the action $\boldsymbol{u}_t$, the reward q_t (instantaneous outflow) is observed, and a new observation $\boldsymbol{x}_{t+1}$ is obtained. Then, in line 13, the experience tuple $(\boldsymbol{x}_t, \boldsymbol{u}_t, q_t, \boldsymbol{x}_{t+1})$ is stored in a buffer $\mathcal{B}$. From lines 14 to 17, the main DNN's parameters θ are optimized. Specifically,

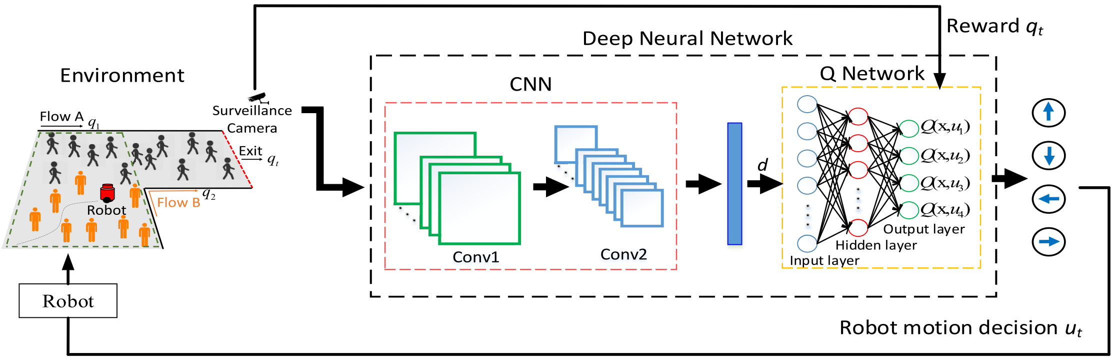


Fig. 2. Overall diagram of the robot motion planning for pedestrian regulation. The DRL-based approach gets the image from the simulated environment, observes the reward q_t accordingly, and generates the robot motion decision $\boldsymbol{u}_t$. The variable $\boldsymbol{d}$ represents the feature vector extracted by the CNN.

in line 14, a batch of tuples $\{(\boldsymbol{x}_j, \boldsymbol{u}_j, q_j, \boldsymbol{x}_{j+1})\}_{j=1}^{D}$ is randomly sampled from the buffer $\mathcal{B}$. With these tuples, in line 15, the target action-value y_j is calculated as

$$y_j = q_j + \gamma Q\left(\boldsymbol{x}_{j+1}, \operatorname*{argmax}_{\boldsymbol{u}} Q\big(\boldsymbol{x}_{j+1}, \boldsymbol{u}; \theta_t\big); \bar{\theta}\right). \tag{7}$$

Then, in line 16, we can derive the loss function as

$$L(\theta_t) = \frac{1}{D}\sum_{j=1}^{D}\big[y_j - Q\big(\boldsymbol{x}_j, \boldsymbol{u}_j; \theta_t\big)\big]^2 \tag{8}$$

which is the mean squared error between the target action-value y_j and the action-value $Q(\boldsymbol{x}_j, \boldsymbol{u}_j; \theta_t)$ estimated by the main DNN. Then, the loss function is minimized by updating the main DNN's parameters θ according to the gradient descent rule, that is, $\theta_{t+1} = \theta_t - \eta \bigtriangledown_{\theta_t} L(\theta_t)$, where η denotes the learning rate and $\bigtriangledown_{\theta_t} L(\theta_t)$ is the gradient of the loss function. Then, at every S step, the target DNN's parameters are reset as $\bar{\theta} = \theta$. After the training process, the main DNN's parameters θ will be outputted for the online robot motion planner deployment.

C. Robot Motion Planner

After the training process in Algorithm 1, the parameters of the DNN will be fixed for the online robot motion planner deployment.

The deployment algorithm is summarized in Algorithm 2. Its input is the images of the simulated environment, and its output is the robot motion decisions. In line 1 of Algorithm 2, we first load the DNN's parameters θ trained by Algorithm 1. Then, the robot position $\boldsymbol{p}_0$ can be initialized as any position in the robot workspace. In the loop starting from line 3, the robot is controlled by the DNN to regulate the pedestrians for T time steps. The image of the environment is obtained from the simulated environment. Then, this image is fed into the CNN to extract discriminative image features, as introduced in Section IV-A. After that, in line 6, these features are fed into the Q network to calculate the action-value $Q(\boldsymbol{x}_t, \boldsymbol{u}; \theta)$ for all permissible robot motion decisions. Then, in line 7, the decision $\boldsymbol{u}_t$ is selected as $\boldsymbol{u}_t = \operatorname{argmax}_{\boldsymbol{u}\in U} Q(\boldsymbol{x}_t, \boldsymbol{u}; \theta)$. Finally, $\boldsymbol{u}_t$ is outputted to the robot.

Algorithm 2 Robot Motion Planning

```
Input: Images of the environment
Output: Robot motion decisions $\boldsymbol{u}_{1:T}$
1: Load the DNN's parameters $\theta$ trained by Algorithm 1.
2: Initialize robot position $\boldsymbol{p}_0$.
3: for Time step $t$ = 1:T do
4:     Obtain the image of the environment $\boldsymbol{x}_t$.
5:     CNN extracts features from the image.
6:     Q network calculates action-value $Q(\boldsymbol{x}_t, \boldsymbol{u}; \theta)$.
7:     $\boldsymbol{u}_t \longleftarrow \operatorname{argmax}_{\boldsymbol{u}\in U} Q(\boldsymbol{x}_t, \boldsymbol{u}; \theta)$
8:     Output robot motion decision $\boldsymbol{u}_t$.
9: end for
```

V. Simulation Results

The effectiveness of the DRL-based end-to-end robot motion planning approach is verified through the simulation experiments. In this section, we first introduce our simulation setup and then present the HRI characteristic results which serve as the ground truth for algorithm validation. After that, we evaluate the performance of the proposed approach with different robot initial positions and pedestrian inflow conditions. We also evaluate its performance under the scenario where the pedestrian inflow changes.

A. Simulation Setup

1) Environment: A simulation environment is developed for the merging pedestrian flow scenario. The image used to train the DNN is generated from the simulated environment and is shown in Fig. 3. The dimension of this image is 200 × 200. The pedestrians in flow A and flow B are represented by red and blue circles in this simulation environment. The robot is denoted by a black square. The size of the environment is 8 × 8 m with an exit corridor of $w = 4$ m in width, that is, $x \sim [4, 8]$ m and $y \sim [4, 8]$ m. The environmental space is continuous for the pedestrians. The robot moves on the grid points of a regular grid defined in $x \sim [0, 4]$ m and $y \sim [0, 8]$ m. The grid size is 0.2 m. The instantaneous outflow q_t is averaged over 5 most recent measurements taken at every second to reduce the noise of the observation.

2) Pedestrian Motion Simulation: We use the existing social force model (SFM) with HRI forces to simulate the

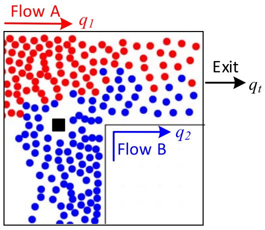
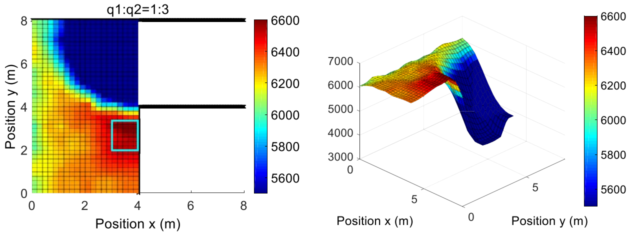


Fig. 3. Snapshot of the environment used in our simulation. The red and blue circles represent the pedestrian in flow A and flow B, respectively. The black square denotes the robot.



(a) (b)

Fig. 4. HRI characteristics for case 1: (a) top-view and (b) 3-D-view. The color indicates the quantity of the accumulated outflow, $\sum_{t=0}^{T} q_t$, at $T = 400$ s. The rectangle in (a) highlights the robot positions with the highest accumulated outflow.

pedestrian's motion and HRI. The SFM is a commonly used model to simulate the pedestrian's behavior and has been used to simulate pedestrian's motion in recent studies on robot navigation [10], [64], [65]. In the SFM, the pedestrian is represented by a plane circle with mass m_i. The motion dynamics of pedestrians are governed by the self-driven force and the interaction forces exerted on a pedestrian by the environment, including other pedestrians, the boundary wall, and the robot. The robot is a passive element in the SFM; that is, the robot does not suffer a repulsive force from the pedestrians. Using the robot as a passive element has been validated by real human–robot experiments [59]. The details of the SFM and its parameters are shown in the supplementary material. The pedestrian's initial speed is set as 2 m/s, and the pedestrian's position is randomly initialized using a uniform distribution $\mathcal{U}(a, b)$. For pedestrians in flow A, $x \sim \mathcal{U}(-5, -3)$ m and $y \sim \mathcal{U}(4.5, 7.5)$ m. For pedestrians in flow B, $x \sim \mathcal{U}(0.5, 3.5)$ m and $y \sim \mathcal{U}(-5, -3)$ m. The radius of the robot is set to 0.2 m.

3) Simulation Scenarios: We have conducted extensive simulations with random robot initial positions and different pedestrian flows. The inflow ratio between flow A and flow B is denoted as q_1/q_2. We set the sum of the number of pedestrians in flow A and flow B to be 300 and vary the inflow ratio q_1/q_2 to create different cases. We first present two simulation cases with different flow ratios q_1/q_2. We set $q_1/q_2 = 1/3$ for case 1 and $q_1/q_2 = 2/1$ for case 2. In each case, we choose different robot initial positions and validate whether the robot can find a good motion planning policy such that the accumulated outflow $\sum_{t=0}^{T} q(t)$ is maximized. The duration of each simulation run is set as $T = 400$ s. Then, we perform the simulation of case 3 where the initial inflow ratio of flow A and flow B is $q_1/q_2 =$ 5:1, and then it changes to $q_1/q_2 =$ 1:11 at $t = 300$ s. The duration of the simulation run is set to 600 s. The robot is expected to adjust its position accordingly to maximize the accumulated outflow under these two flow ratios.

4) DNN Structure: The filter size of the Conv1 layer in CNN is 8 × 8. The output of the Conv1 layer is 16 feature maps, each of which is of dimension 49 × 49. The filter size of the Conv2 layer in CNN is 4 × 4. The output of the second layer is 32 feature maps, each of which is of dimension 23×23. These feature maps are flattened into a 16 928-D vector and fed into the Q network. The hidden layer and output layer of the Q network have 256 and 4 units, respectively.

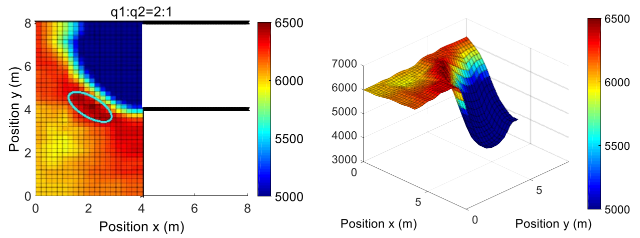
5) Learning Process of DNN: During the learning process, the Adam optimizer is used to minimize the loss function by updating the parameters of DNN. The learning rate is set as 0.0001. The exponential decay rates for the first and the second moment estimates are set as 0.9 and 0.999, respectively. The ReLU activation function is used in the DNN. There are 32 tuples in a batch.

6) Computer Configuration and Implementation: The training and testing of the proposed approach are conducted on a workstation with one 12-core i7-6800K CPU and two NVIDIA TITAN Xp GPUs. The DNN is implemented with Python on GPU 0 while the pedestrian motion model is implemented with PyCUDA on GPU 1.

B. HRI Characteristics

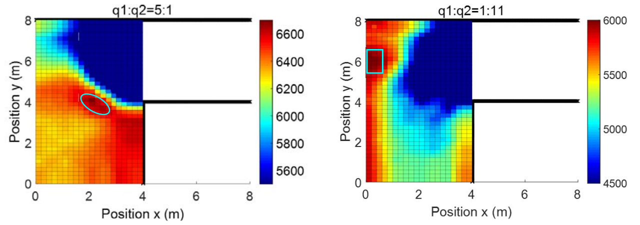
Before verifying the DRL algorithm, we first analyze the effect of the HRI on the pedestrian-accumulated outflow by performing the simulations with the robot placed at different fixed positions. Specifically, we choose a set of 800 robot positions from the region defined in $x \sim [0, 3.8]$ m and $y \sim [0, 7.8]$ m on the grid map with the grid size of 0.2 m. In each run, the robot is placed at one of the positions and the accumulated outflow at $T = 400$ s is recorded. For each position in the set, five runs are repeated and the average accumulated outflow over these five runs is calculated. Fig. 4(a) and (b) illustrates the top view and 3-D view of the HRI characteristics results for case 1, respectively. The quantity of the accumulated outflow of each position is indicated according to the color bar on the right, ranging from dark red (high accumulated outflow) to dark blue (low accumulated outflow). To better show the difference of the positions in the accumulated outflow, we adjust the range of the accumulated outflow value to be mapped on the color map. Thus, accumulated outflow values that are lower than the minimum of the range are indicated by the same color (i.e., dark blue). The robot positions with the highest accumulated outflow are marked by a rectangle in Fig. 4(a). Similarly, the HRI characteristics results for case 2 are presented in Fig. 5. The robot positions with the highest accumulated outflow are marked by an ellipse in Fig. 5(a). The HRI characteristics results for the two inflow ratios in case 3 are illustrated in Fig. 6. The ellipse in Fig. 6(a) and the rectangle in Fig. 6(b) highlight the robot positions with the highest accumulated outflow.

Fig. 4 demonstrates that under the inflow ratio $q_1/q_2 =$ 1:3 (which represents the main flow A being less than the branch



(a) (b)

Fig. 5. HRI characteristics for case 2: (a) top-view and (b) 3-D-view. The color indicates the quantity of the accumulated outflow, $\sum_{t=0}^{T} q_t$, at $T = 400$ s. The ellipse in (a) highlights the robot positions with the highest accumulated outflow.



(a) (b)

Fig. 6. HRI characteristics for case 3: (a) $q_1/q_2 = 5$:1 and (b) $q_1/q_2 =$ 1:11. The color indicates the quantity of the accumulated outflow, $\sum_{t=0}^{T} q_t$, at $T = 400$ s. The ellipse in (a) and the rectangular in (b) highlight the robot positions with the highest accumulated outflow.

flow B), the best position for the robot to stay is the region located on the right of the branch channel (flow B). Fig. 5 shows that under the inflow ratio $q_1/q_2 = 2$:1, the best position for the robot is the region where the two flows merge. A similar region of the best robot positions can be found in Fig. 6(a) for an inflow ratio of $q_1/q_2 = 5$:1. Comparing Fig. 4(a) with Fig. 5(a), we can observe that the optimal regions for case 1 and case 2 are different. It is intuitive that when the main flow A is bigger in case 2, congestion may occur in the merging area, and the robot should stay in the position to impede more pedestrians from getting into the bottleneck to prevent congestion. On the contrary, when branch flow B is bigger in case 1, the robot should stay away from the middle of the flow to have more people passing through to maximize the overall pedestrian outflow. It is worth noting that when the branch flow B keeps getting bigger, the inflow condition can reach the extent that nearly all of the incoming pedestrians are in flow B, which is represented by the inflow ratio of $q_1/q_2 = 1$:11 in case 3. Fig. 6(b) shows that in this case, the robot should stay away from flow B, thus keeping flow B as smooth as possible. The HRI characteristics show that optimal robot positions exist to maximize the outflow of pedestrians and, thus, are used as the "ground-truth" to validate whether the proposed approach can learn the optimal robot motion planning such that the accumulated outflow is maximized under different inflow ratios.

C. Training Process

We present the training performance by running Algorithm 1 in this section. The DNN is trained for 3500 epochs to learn the optimal robot motion policy. In each epoch, the inflow ratio q_1/q_2 is randomly chosen from the set (5:1, 2:1, 1:1, 1:2, 1:3, 1:11), which represents different pedestrian distributions in the two merging flows. The robot initial position is randomly chosen from the grid point. The simulation time for each epoch is 400 s. The evolution of the accumulated outflow at $T = 400$ s over 3500 epochs is shown in Fig. 7. In the first 250 epochs, the robot motion decision is randomly chosen from the four permissible directions. Then, from epoch 250 to epoch 500, the robot motion decision is randomly selected with probability ϵ, otherwise, and it is generated by the proposed DNN. In this phase, the probability ϵ gradually reduces from 1.0 to 0.1 and keeps 0.1 afterward. In Fig. 7, we can observe that the accumulated outflow increases steadily before 2000 epochs, and then it converges with small oscillations. This training result shows that the proposed approach succeeds in learning a robot motion policy to maximize the accumulated outflow under different robot initial positions and different inflow ratios.

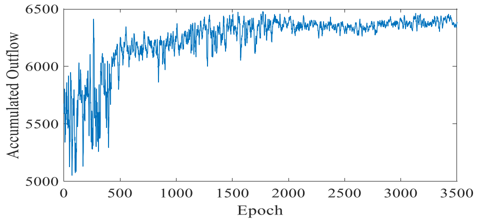
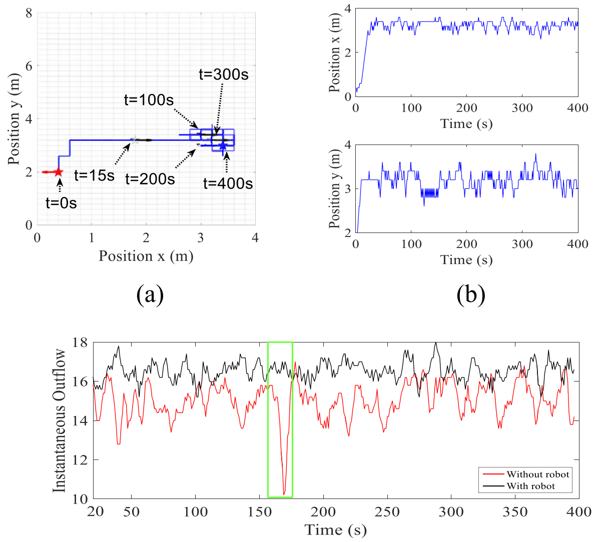


Fig. 7. Evolution of the accumulated outflow at $T = 400$ s over 3500 epochs during the training process.

D. DRL Control for Merging Pedestrian Flows

After the training process, we run Algorithm 2 and present the robot motion planning results in this section. The proposed approach is evaluated under case 1 and case 2 defined in Section V-A. In each case, we conducted extensive simulations with random initial robot positions.

1) Case 1: In case 1, we test the pedestrian flow $q_1/q_2 =$ 1:3. The results of case 1 with the robot initial position [0.4, 2] are illustrated in Fig. 8. The robot trajectory is shown in Fig. 8(a), where the robot moves in the grid-based map defined in the robot's workspace. The robot initial position is denoted by a red star, and the robot positions at different time steps are represented by black stars. The arrow at the star indicates the robot motion direction generated by the proposed approach. The time history of the robot position in the x and y directions is presented in Fig. 8(b). We can observe that after around 20 s, the robot converges into a region, $x \sim [2.6, 3.6]$ and $y \sim [2.6, 3.8]$, which is the optimal region as verified in HRI characteristics and marked by the rectangle in Fig. 4(a). Fig. 8(c) shows the instantaneous outflow q_t where the black curve shows the result with the proposed regulation strategy, and the red curve shows the result without robot regulation. One can see that the instantaneous outflow, q_t, with the proposed regulation is higher than that without robot regulation. Furthermore, in the green box, there is a significant drop in the red curve, which is caused by congestion without the robot regulation. It can be seen that the congestion is avoided by the proposed robot regulation, and the black

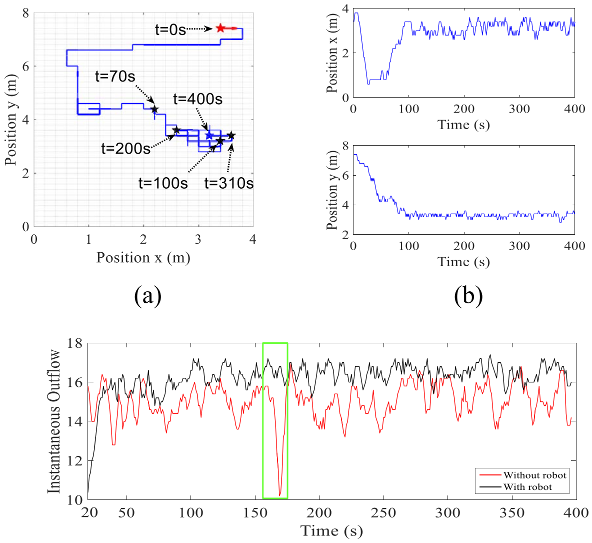


(a)

(b)

(c)

Fig. 8. Robot initial position [0.4, 2] for case 1: (a) robot trajectory; (b) time history of the robot position in the x and y directions; and (c) instantaneous outflow, q_t. The grid in (a) shows the robot workspace. The green box in (c) indicates that the congestion is avoided with robot regulation.



(a)

(b)

(c)

Fig. 9. Robot initial position [3.4, 7.4] for case 1: (a) robot trajectory; (b) time history of the robot position in the x and y directions; and (c) instantaneous outflow, q_t. The grid in (a) shows the robot workspace. The green box in (c) indicates that the congestion is avoided with robot regulation.

curve is relatively more smooth without sharp drops. The accumulated outflow $\sum_{t=0}^{T} q_t$ at $T = 400$ s with and without the robot is 6522 and 5950, respectively. With robot regulation, the accumulated outflow increases by 9.61%.

Similarly, Fig. 9 illustrates the results of case 1 with the robot initial position [3.4, 7.4]. We can see that the robot learns to avoid the region in Fig. 4(a) that results in a low accumulated outflow. After around 100 s, the robot converges into the optimal region. The accumulated outflow at $T = 400$ s with and without the robot is 6313 and 5950, respectively.

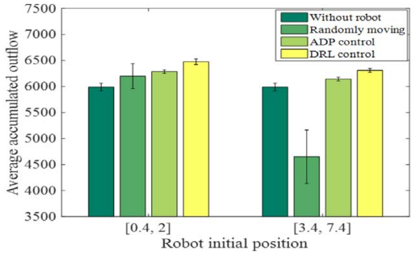
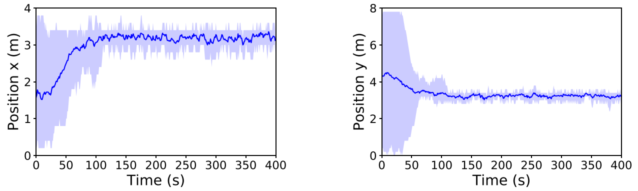


Fig. 10. Average accumulated outflow for case 1 under robot initial positions [0.4, 2] and [3.4, 7.4] over ten runs. Error bar indicates the standard deviation value.



(a)

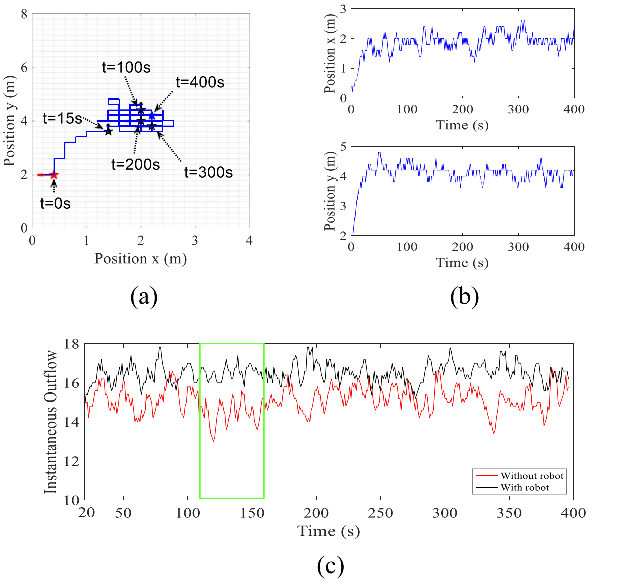
(b)

Fig. 11. Time history of the robot position under ten robot initial positions for case 1: (a) x direction and (b) y direction. The shadow area shows the boundary of these ten trajectories, and the blue solid line illustrates the average position of these trajectories.

With robot regulation, the accumulated outflow increases by 6.10%.

We also provide statistical results to demonstrate the effectiveness of the proposed approach. The performance of the proposed approach is compared with no robot, a randomly moving robot, and an ADP-control robot. For the randomly moving robot, robot motion is randomly selected from the permissible actions. For the ADP-control robot, robot motion is generated by ADP which has been successfully used in [10] for pedestrian regulation. The simulation is repeated ten times for each initial position. The average accumulated outflow at $T = 400$ s is calculated over ten runs. The simulation results for case 1 under robot initial positions [0.4, 2] and [3.4, 7.4] are shown in Fig. 10. Without the robot, the average accumulated outflow is 5989, and its standard deviation is 73. With a randomly moving robot, the average accumulated outflow under these two initial positions is 6198 and 4650, respectively. Their standard deviations are 240 and 514, respectively. With the ADP-control robot, the average accumulated outflow under these two initial positions is 6286 and 6143, respectively. Their standard deviations are 35 and 37, respectively. With the proposed DRL control, the average accumulated outflow increases to 6475 and 6310, respectively. Their standard deviations are 53 and 36, respectively. We can see that the proposed approach greatly improves the accumulated outflow compared to the no-robot case, the randomly moving robot case, and the ADP-control robot case.

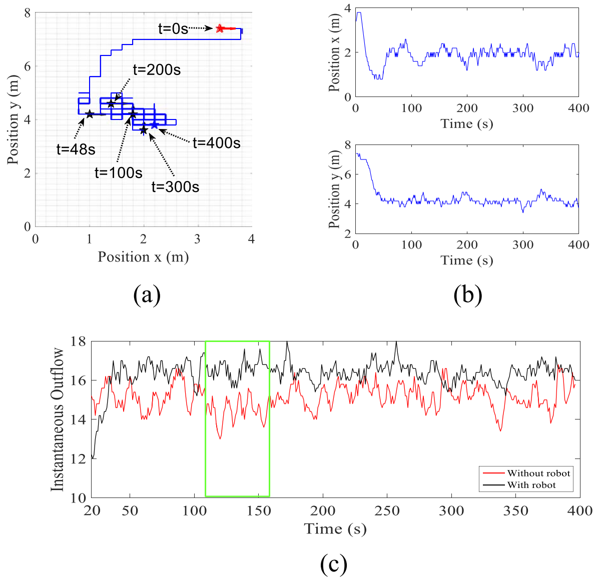
In order to extensively evaluate the performance of the proposed approach, we present the results of ten different robot initial positions in Fig. 11 for case 1. These initial positions are randomly selected from the boundary of the robot workspace, which are away from the optimal region marked in Fig. 4(a).



(a) (b)

(c)

Fig. 12. Robot initial position [0.4, 2] for case 2: (a) robot trajectory; (b) time history of the robot position in the x and y directions; and (c) instantaneous outflow, q_t. The grid in (a) shows the robot workspace. The green box in (c) indicates that the congestion is avoided with robot regulation.



(a) (b)

(c)

Fig. 13. Robot initial position [3.4, 7.4] for case 2: (a) robot trajectory; (b) time history of the robot position in the x and y directions; and (c) instantaneous outflow, q_t. The grid in (a) shows the robot workspace. The green box in (c) indicates that the congestion is avoided with robot regulation.

The shadow area highlights the position range from the maximum to minimum of the ten trajectories at each time step, and the blue solid line illustrates the average position of these trajectories at each time step. We can observe that the robot can converge to the optimal region from different initial positions.

2) Case 2: The results of case 2 with the robot initial position [0.4, 2] are shown in Fig. 12. Fig. 12(a) and (b) shows that the robot converges into the optimal region marked by an ellipse in Fig. 5(a) after around 20 s. One can see from Fig. 12(c) that the instantaneous outflow with robot regulation is higher than that without robot regulation. In the green box, there are sharp drops in the red curve that indicates severe crowd congestion. The accumulated outflow with and without a robot is 6509 and 5989, respectively. With robot regulation, the accumulated outflow increases by 8.68%. Similarly, the results of case 2 with the robot initial position [3.4, 7.4] are shown in Fig. 13. From Fig. 13(a), one can observe that the robot learns to avoid the region in Fig. 5(a) that results in low outflow. In addition, the robot converges into the optimal region after around 50 s. Fig. 13(c) shows the improvement of instantaneous flow. The accumulated outflow with and without a robot is 6356 and 5989, respectively. With robot regulation, the accumulated outflow increases by 6.13%.

The statistical results for case 2 under robot initial positions [0.4, 2] and [3.4, 7.4] are shown in Fig. 14. The simulation is repeated 10 times for each initial position. The average accumulated outflow at $T = 400$ s is calculated over ten runs. Without a robot, the average accumulated outflow is 6027, and its standard deviation is 33. With a randomly moving robot, the average accumulated outflow under these two initial positions is 6060 and 4796, respectively. Their standard deviations are 69 and 821, respectively. With an ADP-control robot, the average accumulated outflow under these two initial positions is 6223 and 6129, respectively. Their standard deviations are 27 and 18, respectively. With the proposed DRL control, the average accumulated outflow increases to 6485 and 6347, respectively. Their standard deviations are 17 and 30, respectively. We can see that the proposed approach greatly improves the accumulated outflow compared to these benchmark solutions.

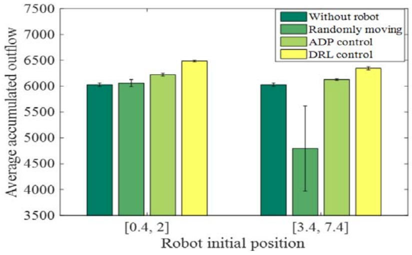
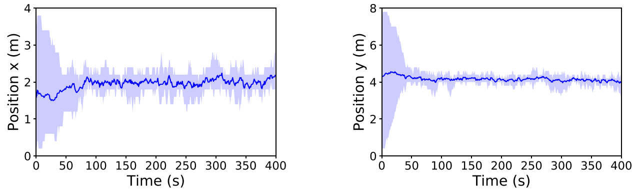


Fig. 14. Average accumulated outflow for case 2 under robot initial positions [0.4, 2] and [3.4, 7.4] over ten runs. Error bar indicates the standard deviation value.



(a) (b)

Fig. 15. Time history of the robot position under ten robot initial positions for case 2: (a) x direction and (b) y direction. The shadow area shows the boundary of these ten trajectories, and the blue solid line shows the average position of these trajectories.

We also present the results of ten robot initial positions in Fig. 15 for case 2. These initial positions are randomly selected from the boundary of the robot workspace, which are away

TABLE I
COMPUTATIONAL TIME AND RESOURCE USAGE

	Time	GPU 0 memory	GPU 1 memory	CPU memory
Training	30 hours	4.44%	2.64%	2.07%
Online	125 ms	4.44%	2.64%	1.49%

from the optimal region marked in Fig. 5(a). The shadow area highlights the position range from the maximum to minimum of the ten trajectories at each time step, and the solid blue line illustrates the average position of these trajectories at each time step. We can observe that the robot can converge to the optimal region marked in Fig. 5(a) from different initial positions.

These results demonstrate that the proposed approach succeeds in learning the optimal robot motion planning such that the accumulated outflow is maximized under different inflow ratios and different robot initial positions.

3) Computational Time and Resource Usage: The computational time and resource usage for the training phase and the testing phase are presented in Table I. It takes about 30 h to train the DNN on the workstation mentioned in Section V-A. The GPU memory usage is 4.44% and 2.64% for GPU 0 and GPU 1, respectively. The CPU memory usage is 2.07%. For the online robot motion planner running Algorithm 2, the image of the environment is fed into the robot every 1 s. It takes about 125 ms for Algorithm 2 to generate the robot motion decision after receiving the input image. The GPU memory usage is 4.44% and 2.64%. The CPU memory usage is 1.49%. We can see that after the training, the robot motion planner is fast enough for online control.

E. DRL Control for Changing Pedestrian Flows

In this section, we evaluate the performance of the proposed DRL-based approach when the pedestrian inflow ratio changes. We conduct the simulation of case 3 where the initial inflow ratio of flow A and flow B is $q_1/q_2 = 5{:}1$, and then it changes to $q_1/q_2 = 1{:}11$ at $t = 300$ s. Case 3 represents the scenario where the main inflow (flow A) is more than the branch inflow (flow B) initially and then the main inflow receives less than the branch inflow. The robot is expected to adjust its position accordingly to maximize the accumulated outflow under these two flow ratios. Fig. 6(a) and (b) shows the HRI characteristics of the inflow ratios $q_1/q_2 = 5{:}1$ and $q_1/q_2 = 1{:}11$, respectively. The ellipse in Fig. 6(a) and the rectangle in Fig. 6(b) highlight the robot positions with the highest accumulated outflow. The robot is expected to converge into the ellipse region in Fig. 6(a) for $q_1/q_2 = 5{:}1$ and then adjust to the rectangular region in Fig. 6(b) for $q_1/q_2 = 1{:}11$.

Fig. 16 shows the simulation results for case 3 where the robot initial position is [1, 1]. We can observe that the robot converges to the ellipse region in Fig. 6(a) at about 30 s. When the inflow ratio changes at 300 s, the robot adjusts its position and converges to the rectangular region in Fig. 6(b) after 340 s. These results verify that the proposed approach can replan the robot motion in real time according to the change of the pedestrian inflows.

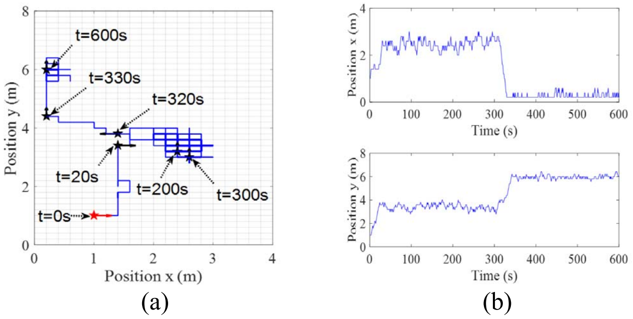


Fig. 16. DRL control for changing pedestrian flows: (a) robot trajectory and (b) time history of the robot position in the x and y directions. The grid in (a) shows the robot workspace.

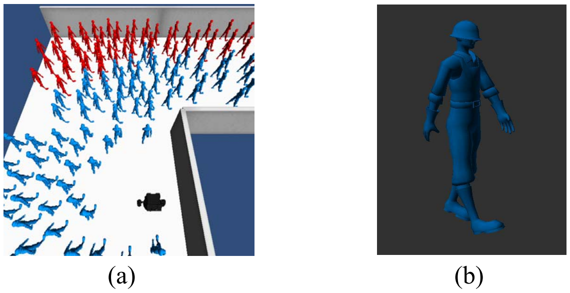
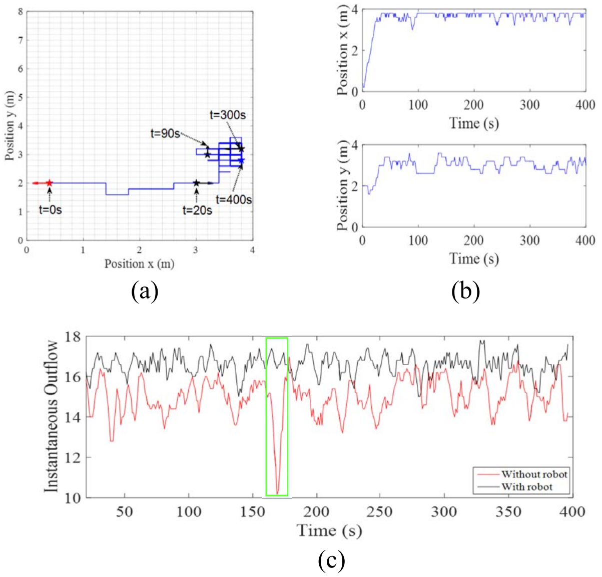


Fig. 17. 3-D environment constructed by the Unity 3-D engine. (a) Snapshot of the 3-D environment. (b) 3-D pedestrian model.

VI. EVALUATION RESULTS IN 3-D ENVIRONMENT

In this section, we have evaluated our proposed approach in a 3-D continuous environment constructed by the Unity 3-D engine. The new 3-D environment is shown in Fig. 17(a). The pedestrians in flow A and flow B are represented by red and blue, respectively. The 3-D pedestrian model is shown in Fig. 17(b).

The evaluation results under pedestrian flow $q_1/q_2 = 1{:}3$ are shown in Fig. 18. The robot trajectory is shown in Fig. 18(a). The robot initial position is denoted by a red star, and the robot positions at different time steps are represented by black stars. The arrow at the star indicates the robot motion direction generated by the proposed approach. The time history of the robot position in the x and y directions is presented in Fig. 18(b). We can observe that after around 20 s, the robot converges into a region, $x \sim [2.8, 3.8]$ and $y \sim [2.0, 3.6]$, which is in the optimal region as verified in HRI characteristics and marked by the rectangle in Fig. 4(a). Fig. 18(c) shows the instantaneous outflow, q_t, where the black curve shows the result with the proposed regulation strategy, and the red curve shows the result without robot regulation. One can see that the instantaneous outflow q_t with the proposed regulation is higher than that without robot regulation. Furthermore, in the green box, there is a significant drop in the red curve, which is caused by congestion without robot regulation. It can be seen that congestion is avoided by the proposed robot regulation, and the black curve is relatively more smooth without sharp drops. The accumulated outflow $\sum_{t=0}^{T} q_t$ at $T = 400$ s with and without the robot are 6549 and 5950, respectively. With the proposed robot regulation, the accumulated outflow



(a)

(b)

(c)

Fig. 18. Evaluation results in the 3-D environment: (a) robot trajectory; (b) time history of the robot position in the x and y directions; and (c) instantaneous outflow, q_t. The grid in (a) shows the robot workspace. The green box in (c) indicates that the congestion is avoided with robot regulation.

increases by 10.07%. These results verify the effectiveness of our proposed approach in this 3-D environment.

VII. Conclusion

In this paper, we proposed a DRL-based approach for robot motion planning to regulate pedestrian flows. The robot motion planning problem was solved using a DNN that consists of a CNN and a Q network to learn the optimal policy for robot motion decisions that maximize the pedestrian outflow. The proposed approach provides an end-to-end motion planner that directly uses the images of the environment. In comparison with existing work on learning-based pedestrian regulation, the CNN is applied to extract discriminative features from the input images for optimal robot motion decisions. Extensive simulations were performed, and the results verified the effectiveness of the proposed approach on pedestrian flow regulation.

This paper focused on the formulation of the robot-assisted pedestrian regulation problem and the novel DRL approach to solve it. While our approach was validated in a simulated environment in this paper, we plan to implement the proposed approach in real-world scenarios in our future work. Specifically, we plan to set up an environment with an installed surveillance camera and a pedestrian tracking system, collect real images of pedestrian flows from the surveillance camera, transmit images and instantaneous outflow data to a mobile robot via a local-area network, and have the robot calculate motion decisions using our proposed algorithm.

References

[1] P. B. Luh, C. T. Wilkie, S.-C. Chang, K. L. Marsh, and N. Olderman, “Modeling and optimization of building emergency evacuation considering blocking effects on crowd movement,” *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 4, pp. 687–700, Oct. 2012.

[2] E. Boukas, I. Kostavelis, A. Gasteratos, and G. C. Sirakoulis, “Robot guided crowd evacuation,” *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 739–751, Apr. 2015.

[3] B. Tang, C. Jiang, H. He, and Y. Guo, “Human mobility modeling for robot-assisted evacuation in complex indoor environments,” *IEEE Trans. Human–Mach. Syst.*, vol. 46, no. 5, pp. 694–707, Oct. 2016.

[4] D. Helbing, L. Buzna, A. Johansson, and T. Werner, “Self-organized pedestrian crowd dynamics: Experiments, simulations, and design solutions,” *Transp. Sci.*, vol. 39, no. 1, pp. 1–24, 2005.

[5] G. A. Frank and C. O. Dorso, “Room evacuation in the presence of an obstacle,” *Physica A Stat. Mech. Appl.*, vol. 390, no. 11, pp. 2135–2145, 2011.

[6] Y. Zhao *et al.*, “Optimal layout design of obstacles for panic evacuation using differential evolution,” *Physica A Stat. Mech. Appl.*, vol. 465, pp. 175–194, Jan. 2017.

[7] K. Yamamoto and M. Okada, “Control of swarm behavior in crossing pedestrians based on temporal/spatial frequencies,” *Robot. Auton. Syst.*, vol. 61, no. 9, pp. 1036–1048, 2013.

[8] B. D. Eldridge and A. A. Maciejewski, “Using genetic algorithms to optimize social robot behavior for improved pedestrian flow,” in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 1, 2005, pp. 524–529.

[9] J. A. Kirkland and A. A. Maciejewski, “A simulation of attempts to influence crowd dynamics,” in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, vol. 5, 2003, pp. 4328–4333.

[10] C. Jiang, Z. Ni, Y. Guo, and H. He, “Learning human–robot interaction for robot-assisted pedestrian flow optimization,” *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2017.2725300.

[11] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, “Human-aware robot navigation: A survey,” *Robot. Auton. Syst.*, vol. 61, no. 12, pp. 1726–1743, 2013.

[12] H. Kidokoro, T. Kanda, D. Bršćić, and M. Shiomi, “Simulation-based behavior planning to prevent congestion of pedestrians around a robot,” *IEEE Trans. Robot.*, vol. 31, no. 6, pp. 1419–1431, Dec. 2015.

[13] P. Trautman, J. Ma, R. M. Murray, and A. Krause, “Robot navigation in dense human crowds: The case for cooperation,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2013, pp. 2153–2160.

[14] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, “Socially compliant mobile robot navigation via inverse reinforcement learning,” *Int. J. Robot. Res.*, vol. 35, no. 11, pp. 1289–1307, 2016.

[15] H. Modares, I. Ranatunga, F. L. Lewis, and D. O. Popa, “Optimized assistive human–robot interaction using reinforcement learning,” *IEEE Trans. Cybern.*, vol. 46, no. 3, pp. 655–667, Mar. 2016.

[16] D. McColl, C. Jiang, and G. Nejat, “Classifying a person’s degree of accessibility from natural body language during social human–robot interactions,” *IEEE Trans. Cybern.*, vol. 47, no. 2, pp. 524–538, Feb. 2017.

[17] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[18] D. Maturana and S. Scherer, “VoxNet: A 3D convolutional neural network for real-time object recognition,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 922–928.

[19] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, “Multimodal deep learning for robust RGB-D object recognition,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 681–687.

[20] G. L. Oliveira, A. Valada, C. Bollen, W. Burgard, and T. Brox, “Deep learning for human part discovery in images,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2016, pp. 1634–1641.

[21] R. Hadsell *et al.*, “Deep belief net learning in a long-range vision system for autonomous off-road driving,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 628–633.

[22] A. Giusti *et al.*, “A machine learning approach to visual perception of forest trails for mobile robots,” *IEEE Robot. Autom. Lett.*, vol. 1, no. 2, pp. 661–667, Jul. 2016.

[23] L. Tai, S. Li, and M. Liu, “A deep-network solution towards model-less obstacle avoidance,” in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2016, pp. 2759–2764.

[24] I. Lenz, H. Lee, and A. Saxena, “Deep learning for detecting robotic grasps,” *Int. J. Robot. Res.*, vol. 34, nos. 4–5, pp. 705–724, 2015.

[25] S. Levine, N. Wagener, and P. Abbeel, “Learning contact-rich manipulation skills with guided policy search,” in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 156–163.

[26] S. Levine, C. Finn, T. Darrell, and P. Abbeel, “End-to-end training of deep visuomotor policies,” *J. Mach. Learn. Res.*, vol. 17, no. 39, pp. 1–40, 2016.

[27] P. Trautman, J. Ma, R. M. Murray, and A. Krause, "Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 335–356, 2015.

[28] D. Brščić, T. Kanda, T. Ikeda, and T. Miyashita, "Person tracking in large public spaces using 3-D range sensors," *IEEE Trans. Human–Mach. Syst.*, vol. 43, no. 6, pp. 522–534, Nov. 2013.

[29] M. Luber, L. Spinello, J. Silva, and K. O. Arras, "Socially-aware robot navigation: A learning approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 902–907.

[30] B. Kim and J. Pineau, "Socially adaptive path planning in human environments using inverse reinforcement learning," *Int. J. Soc. Robot.*, vol. 8, no. 1, pp. 51–66, 2016.

[31] Z. Wan, X. Hu, H. He, and Y. Guo, "A learning based approach for social force model parameter estimation," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 4058–4064.

[32] H. R. Beom and H. S. Cho, "A sensor-based navigation for a mobile robot using fuzzy logic and reinforcement learning," *IEEE Trans. Syst., Man, Cybern.*, vol. 25, no. 3, pp. 464–477, Mar. 1995.

[33] M. Asada, S. Noda, S. Tawaratsumida, and K. Hosoda, "Purposive behavior acquisition for a real robot by vision-based reinforcement learning," *Mach. Learn.*, vol. 23, nos. 2–3, pp. 279–303, May 1996.

[34] C. Gaskett, L. Fletcher, and A. Zelinsky, "Reinforcement learning for a vision based mobile robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, vol. 1, 2000, pp. 403–409.

[35] Y. Duan, B. Cui, and H. Yang, "Robot navigation based on fuzzy RL algorithm," in *Proc. Adv. Neural Netw. (ISNN)*, 2008, pp. 391–399.

[36] H. Mhaskar, Q. Liao, and T. A. Poggio, "When and why are deep networks better than shallow ones?" in *Proc. Assoc. Adv. Artif. Intell.*, 2017, pp. 2343–2349.

[37] Z. Zhang, D. Zhao, J. Gao, D. Wang, and Y. Dai, "FMRQ—A multiagent reinforcement learning algorithm for fully cooperative tasks," *IEEE Trans. Cybern.*, vol. 47, no. 6, pp. 1367–1379, Jun. 2017.

[38] Q. Zhang and D. Zhao, "Data-based reinforcement learning for nonzero-sum games with unknown drift dynamics," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2018.2830820.

[39] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[40] P. Rodriguez *et al.*, "Deep pain: Exploiting long short-term memory networks for facial expression classification," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2017.2662199.

[41] Z. Wan, H. He, and B. Tang, "A generative model for sparse hyperparameter determination," *IEEE Trans. Big Data*, vol. 4, no. 1, pp. 2–10, Mar. 2018.

[42] L. Wu, Y. Wang, X. Li, and J. Gao, "Deep attention-based spatially recursive networks for fine-grained visual recognition," *IEEE Trans. Cybern.*, to be published, doi: 10.1109/TCYB.2018.2813971.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[44] Z. Wan and H. He, "Weakly supervised object localization with deep convolutional neural network based on spatial pyramid saliency map," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2017, pp. 4177–4181.

[45] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[46] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.

[47] Y. He *et al.*, "Deep-reinforcement-learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10433–10445, Nov. 2017.

[48] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 285–292.

[49] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 31–36.

[50] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard, "Deep reinforcement learning with successor features for navigation across similar environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2017, pp. 2371–2378.

[51] Y. Zhu *et al.*, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2017, pp. 3357–3364.

[52] F. Martinez-Gil, M. Lozano, and F. Fernández, "MARL-Ped: A multi-agent reinforcement learning based framework to simulate pedestrian groups," *Simulat. Modell. Pract. Theory*, vol. 47, pp. 259–275, Sep. 2014.

[53] L. Torrey, "Crowd simulation via multi-agent reinforcement learning," in *Proc. 6th AAAI Conf. Artif. Intell. Interact. Digit. Entertainment*, 2010, pp. 89–94.

[54] D. Helbing, A. Johansson, and H. Z. Al-Abideen, "Dynamics of crowd disasters: An empirical study," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top*, vol. 75, no. 4, 2007, Art. no. 046109.

[55] A. Johansson, D. Helbing, H. Z. Al-Abideen, and S. Al-Bosta, "From crowd dynamics to crowd safety: A video-based analysis," *Adv. Complex Syst.*, vol. 11, no. 4, pp. 497–527, 2008.

[56] Y. Tajima and T. Nagatani, "Clogging transition of pedestrian flow in T-shaped channel," *Physica A Stat. Mech. Appl.*, vol. 303, nos. 1–2, pp. 239–250, 2002.

[57] J. Zhang, W. Klingsch, A. Schadschneider, and A. Seyfried, "Transitions in pedestrian fundamental diagrams of straight corridors and T-junctions," *J. Stat. Mech. Theory Exp.*, vol. 2011, no. 6, 2011, Art. no. P06004.

[58] B. Haworth *et al.*, "Using synthetic crowds to inform building pillar placements," in *Proc. IEEE Virtual Humans Crowds Immersive Environ.*, 2016, pp. 7–11.

[59] Z. Chen, C. Jiang, and Y. Guo, "Pedestrian-robot interaction experiments in an exit corridor," in *Proc. 15th Int. Conf. Ubiquitous Robots*, Honolulu, HI, USA, Jun. 2018, pp. 29–34.

[60] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv:1603.07285*, Jan. 2018.

[61] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Math. Control Signals Syst.*, vol. 2, no. 4, pp. 303–314, 1989.

[62] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. Assoc. Adv. Artif. Intell.*, 2016, pp. 2094–2100.

[63] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, vol. 1. Cambridge, MA, USA: MIT Press, 1998.

[64] G. Ferrer, A. Garrell, and A. Sanfeliu, "Robot companion: A social-force based approach with human awareness-navigation in crowded environments," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Nov. 2013, pp. 1688–1694.

[65] D. Vasquez, B. Okal, and K. O. Arras, "Inverse reinforcement learning algorithms and features for robot navigation in crowds: An experimental comparison," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Sep. 2014, pp. 1341–1346.

Zhiqiang Wan (S'16) received the B.S. degree in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2012 and the M.S. degree in electrical engineering from the School of Electrical and Electronics Engineering, Huazhong University of Science and Technology, Wuhan, China, in 2015. He is currently pursuing the Ph.D. degree in electrical engineering with the School of Electrical, Computer and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA.

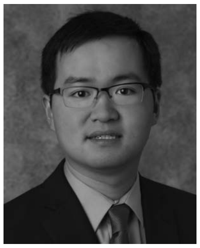
His current research interests include deep learning, robotics, and deep reinforcement learning.

Chao Jiang (S'13) received the B.S. degree in measuring and control technology and instrumentation from Chongqing University, Chongqing, China, in 2009. He is currently pursuing the Ph.D. degree in electrical engineering with the Stevens Institute of Technology, Hoboken, NJ, USA.

He was a Research Assistant with the Key Laboratory of Optoelectronic Technology and Systems, Ministry of Education, Chongqing University, from 2009 to 2012. His current research interests include learning-based robot control, human–robot interaction, deep reinforcement learning, and multirobot cooperative localization.

Muhammad Fahad (S'13) received the B.S. degree in electrical engineering from the University of Engineering and Technology, Lahore, Pakistan, in 2007 and the M.S. degree in electrical engineering from the Stevens Institute of Technology, Hoboken, NJ, USA, in 2013, where he is currently pursuing the Ph.D. degree in electrical engineering with the Robotics and Automation Laboratory.

His current research interests include cooperative distributed localization, environmental monitoring, human–robot interaction, and reinforcement learning.



Zhen Ni (M'15) received the B.S. degree in control science and engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2010 and the Ph.D. degree in electrical, computer and biomedical engineering from the University of Rhode Island, Kingston, RI, USA, in 2015.

He is currently an Assistant Professor with the Department of Electrical Engineering and Computer Science, South Dakota State University, Brookings, SD, USA. His current research interests include computational intelligence, reinforcement learning, and cyber-physical systems.

Prof. Ni was a recipient of the URI Excellence in Doctoral Research Award in 2016, the Chinese Government Award for Outstanding Students Abroad in 2014, and the Second Prize of Graduate Student Poster Contest in the IEEE Power and Energy Society General Meeting in 2015. He has been actively involved in numerous conference and workshop organization committees in the society, including the General Co-Chair of the IEEE CIS Winter School, Washington, DC, USA, in 2016. He has been an Associate Editor of the *IEEE Computational Intelligence Magazine* since 2018, and was a Guest Editor of *IET Cyber-Physical Systems: Theory and Applications* from 2017 to 2018.

Yi Guo (SM'04) received the B.S. and M.S. degrees in electrical engineering from the Xi'an University of Technology, Xi'an, China, in 1992 and 1995, respectively, and the Ph.D. degree in electrical engineering from the University of Sydney, Sydney, NSW, Australia, in 1999.

She was a Postdoctoral Research Fellow with Oak Ridge National Laboratory, Oak Ridge, TN, USA, from 2000 to 2002, and a Visiting Assistant Professor with the University of Central Florida, Orlando, FL, USA, from 2002 to 2005. She joined the Department of Electrical and Computer Engineering, Stevens Institute of Technology, Hoboken, NJ, USA, in 2005, where she is currently a Professor. She has published over 100 peer-reviewed journal and conference papers, authored the book entitled *Distributed Cooperative Control: Emerging Applications* (Wiley, 2017) and edited the book entitled *Micro/Nano-Robotics for Biomedical Applications* (Springer, 2013). Her current research interests include autonomous mobile robotics, distributed sensor networks, and nonlinear control systems.

Dr. Guo currently serves on the editorial boards of several journals, including the *IEEE Robotics and Automation Magazine* and IEEE/ASME TRANSACTIONS ON MECHATRONICS. She served on organizing committees of the IEEE International Conference on Robotics and Automation in 2006, 2008, 2014, and 2015.

Haibo He (SM'11–F'18) received the B.S. and M.S. degrees in electrical engineering from the Huazhong University of Science and Technology, Wuhan, China, in 1999 and 2002, respectively, and the Ph.D. degree in electrical engineering from Ohio University, Athens, OH, USA, in 2006.

He is currently the Robert Haas Endowed Chair Professor with the Department of Electrical, Computer, and Biomedical Engineering, University of Rhode Island, Kingston, RI, USA. He has published one sole-author research book (Wiley), edited one book (Wiley-IEEE), and six conference proceedings (Springer), and has authored and co-authored over 300 peer-reviewed journal and conference papers. His current research interests include computational intelligence, machine learning, data mining, and various applications.

Dr. He was a recipient of the IEEE International Conference on Communications Best Paper Award in 2014, the IEEE CIS Outstanding Early Career Award in 2014, and the National Science Foundation CAREER Award in 2011. He was the General Chair of the IEEE Symposium Series on Computational Intelligence in 2014. He is currently the Editor-in-Chief of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.