Adaptive Environment Modeling Based Reinforcement Learning (AEMCARL) for Collision Avoidance in Crowded Scenes

Shuaijun Wang[1,2], Rui Gao[1], Chengyang Li[1], Shengduo Chen[1], Ruihua Han[1,3], and Qi Hao[1,4,*]

***Abstract*— The major challenges of collision avoidance for robot navigation in crowded scenes lie in accurate environment modeling, fast perceptions, and trustworthy motion planning policies. This paper presents a novel adaptive environment model based collision avoidance reinforcement learning (i.e., AEMCARL) framework for an unmanned robot to achieve collision-free motions in challenging navigation scenarios. The novelty of this work is threefold: (1) developing a hierarchical network of extended gated recurrent units (EGRUs) for environment modeling; (2) developing an adaptive perception mechanism with an attention module; (3) developing a comprehensive reinforcement learning (RL) framework to jointly train the environment model, perception function and motion planning policy. The proposed method is tested with the Gym-Gazebo simulator and a group of robots (Husky and Turtlebot) under various crowded scenes. Both simulation and experimental results have demonstrated the superior performance of the proposed method over baseline methods.**

I. INTRODUCTION

End-to-end RL based unmanned mobile robots are advantageous in the high degree of autonomy, robustness against unmodeled uncertainties, and high-speed decision [1]. The main components of RL-based unmanned robots include (1) sensing, motion and communication units, (2) world and robot models, (3) policy and reward functions, and (4) state and action spaces. Developing RL-based autonomous robot systems for crowded scenes, as shown in Fig. 1, has to deal with the following technical challenges:

1) **Dynamic environment modeling**. In the real world, many objects such as pedestrians, vehicles, animals move around with various behaviors and interactions; how to develop a hierarchical model to represent such a dynamic environment in different degrees of complexity is still a challenging problem;
2) **Adaptive perception mechanism**. The robot should be able to perceive the environment with an attention mechanism, and adaptively use computational resources within a certain degree of perception confidences.

This work is partially supported by the National Natural Science Foundation of China (No: 61773197); the Science and Technology Innovation Committee of Shenzhen City (No: JCYJ20200109141622964); and the Nanshan District Science and Technology Innovation Bureau (No: LHTD20170007); and the Intel ICRI-IACV Research Agreement (Intel CG#52514373).

*Corresponding author: Qi Hao (hao.q@sustech.edu.cn).

[1]Department of Computer Science and Engineering, Southern University of Science and Technology, Shenzhen, Guangdong, China, 518055

[2]Harbin Institute of Technology, Harbin, Heilongjiang, China, 150001

[3]Department of Computer Science, The University of Hongkong, 999077

[4]Rsearch Institute of Trustworthy Autonomous Systems, Southern University of Science and Technology, Shenzhen, Guangdong, China, 518055

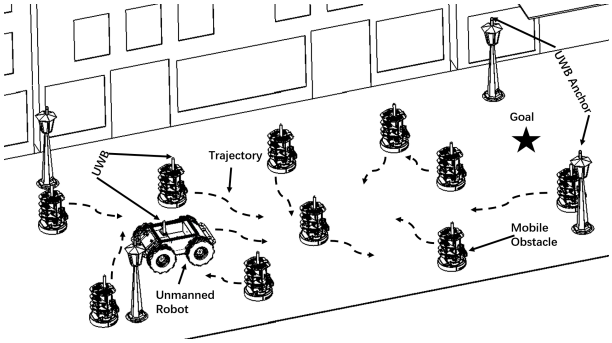

Fig. 1. An illustration of an unmanned robot with the capability of collision avoidance in crowded scenes (of robotic mobile obstacles). All robots are equipped with Ultra-wideband (UWB) modules for distance measurement and localization.

3) **Collision-free policy for motion planning**. The quality of RL-based motion planning policies relies on the selection of reward functions and state representation. Besides, state filtering could also help improve the system performance given partial observations.

Most RL based crowd-aware approaches use simple dynamic environment models such as MLPs which cannot fully describe real-world complexities and uncertainties [2]. Meanwhile, perception functions have to trade off between decision speed and environment modeling accuracy by selecting proper structures of deep learning models [3]–[5]. On the other hand, filtering and attention mechanisms [5], [6] have been developed to make better decisions with reduced computational workloads. However, no such a RL based framework has been developed yet, which contains high order environment models as well as adaptive perception, attention and filtering modules.

In this paper, we propose a robust and high-speed motion planning framework with an adaptive environmental model (AEM) to achieve collision-free navigation in crowded scenarios. The estimated perception confidences are used to change the AEM structure in real time. Attention and filtering modules are developed to improve the generalization capability of the system. The main contributions of this paper include

1) Developing a novel RL-based motion planning framework, which consists of modules of modeling, attention, filtering and action, to achieve real-time navigation with collision avoidance;
2) Developing a hierarchical EGRU model to represent multiple agents of dynamic environments;
3) Developing a perception confidence based adaptation mechanism as well as an attention module to achieve

Manuscript 246 submitted to 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 4, 2021.

TABLE I

A BRIEF SUMMARY OF TRADITIONAL AND REINFORCEMENT LEARNING BASED COLLISION-FREE NAVIGATION METHODS

Scheme	Navigation Method	Dynamic Environment Modeling			Adaptive Perception Mechanism			Collision-free Policy		
		Agent number	Multi-agent representation	Hierarchical representation	Perception confidence	Tuning scheme	Attention mechanism	CA (Learning) scheme	Reward	Filtering
Traditional Approach	RVO [7]	High	Math modeling	×	×	×	×	RVO	N/A	×
	ORCA [8]	High	Math modeling	×	×	×	×	RVO	N/A	×
RL-based Approach	LSTM-RL [4]	High	LSTM & MLP	×	×	×	×	DQN	+	×
	CADRL [9]	Low	MLP	×	×	×	×	DQN	+	×
	SARL [6]	High	MLP	×	×	×	√	DQN	+	×
	RGL [5]	High	GNN	×	×	×	√	DQN	+	×
	AEMCARL (Ours)	High	EGRU	√	√	√	√	DQN	++	√

The number of + reflects the performance of this functionality of the method. Symbol √: This functionality is supported. Symbol ×: This functionality is not supported or not considered. CA: Collision avoidance. GNN: Graph neural network.

fast decision with reduced computational complexity;

4) Developing a set of robust navigation policy training algorithms with a state filtering module and a properly designed reward function. The open-source code is available at https://github.com/SJWang2015/AEMCARL.

The rest of this paper is organized as follows. Section II introduces the related work on RL based motion planning. Section III describes the system setup and problem statement. Section IV presents the proposed method. Section V provides the experiment results and discussions. Section VI concludes this paper and outlines future work.

II. RELATED WORK

TABLE I summarizes a number of navigation methods with collision avoidance for a single robot in crowded scenes. Compared with conventional methods, the RL-based approaches can learn from the various simulation experiences to generate robust and high-efficiency motion planning policies. The most challenging problem for motion planning in crowded scenes is to build dynamic environment models (DEMs) which should accommodate a large number of mobile agents, and describe the spatial and temporal interactions among agents. Most DEMs can be classified into two groups: object-centric and relation-centric. The former emphasizes the pose and velocity information of agents [10]; while the latter describes the current and future relationships among agents [10]. To achieve better modeling performance, fully connected (FC) layers and LSTM [3] units have been used to describe long-term and long-range relation-centric interactions among agents [4], [11]. Compared with LSTM units, gated recurrent units (GRU) [12] are advantageous in high computational efficiency and fast convergence. However, no hierarchical GRU or extended GRU (EGRU) networks have been developed to represent complex dynamic environments with a large number of agents.

Recurrent and feedforward networks [4], [6], [9] with high order structures have been used to filter and model the complex dynamic environments. However, in a typical crowded scene, the number of mobile obstacles which might cause collisions might vary from time to time. The prediction of the interactions among a large number of agents usually incur high computational workloads. Attention mechanisms can help the unmanned robot to focus on most threatening mobile obstacles in the environment, which are not necessarily correlated to the robot-obstacle distance [13]. On the other hand, it is necessary to develop a set of self-tuning mechanisms, in which the environment model structure can be changed according to the perception confidences [14]. Despite many efforts in this aspect have been used for natural language processing [14], little work has been done to achieve online self-tuning DEMs for collision avoidance.

Compared with policy-based and actor-critic RL approaches such as PPO [15] and A3C [16], the value-based off-policy Deep Q-learning network (DQN) [17] are advantageous in simple structure, low computational complexity and fast training convergence. However, it is very critical to choose proper reward functions for DQN based collision-free navigation policies, which should avoid being stuck in freezing points [2] and move forward to the goal position as fast as possible. Besides, proper representations of the interactions of critical agents under attention are also helpful for network training convergence [6], [17] .

In this work, a value-based RL method, AEMCARL, is proposed to achieve collision-free navigation policies. AEMCARL uses a hierarchical environment model with an adaptive self-tuning module to learn the relationships among multiple agents. Attention and filtering modules are also used to improve the computational efficiency and training stability.

III. SYSTEM SETUP AND PROBLEM STATEMENT

A. System Setup

Fig. 2 illustrates the proposed RL-based navigation system with collision avoidance in crowded scenes for an unmanned robot. The experiment platform contains a Husky as the unmanned robot and a group of Turtlebots as mobile obstacles. All robots are equipped with UWB modules, along with stationary UWB anchors, for localization. All the odometry readings, including poses and velocities, UWB readings, and action commands are exchanged through the ROS communication network. The Gym-based simulator [6] is used to train the hierarchical environment model (HEM), adaptive perception module (APM), policy function and evaluation network.

B. Problem Statement

The state of the robot at time t is denoted as s_t, which includes velocity, (v_x^r, v_y^r), position, (x^r, y^r), goal position, (g_x^r, g_y^r), and preferred velocity value, v_r^{pref}. The state of mobile obstacles at time t is denoted as o_t, which includes the velocity, (v_x^o, v_y^o), position, (x^o, y^o), and preferred velocity value, v_o^{pref}. The input of the RL network is the joint state

Manuscript 246 submitted to 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received March 4, 2021.

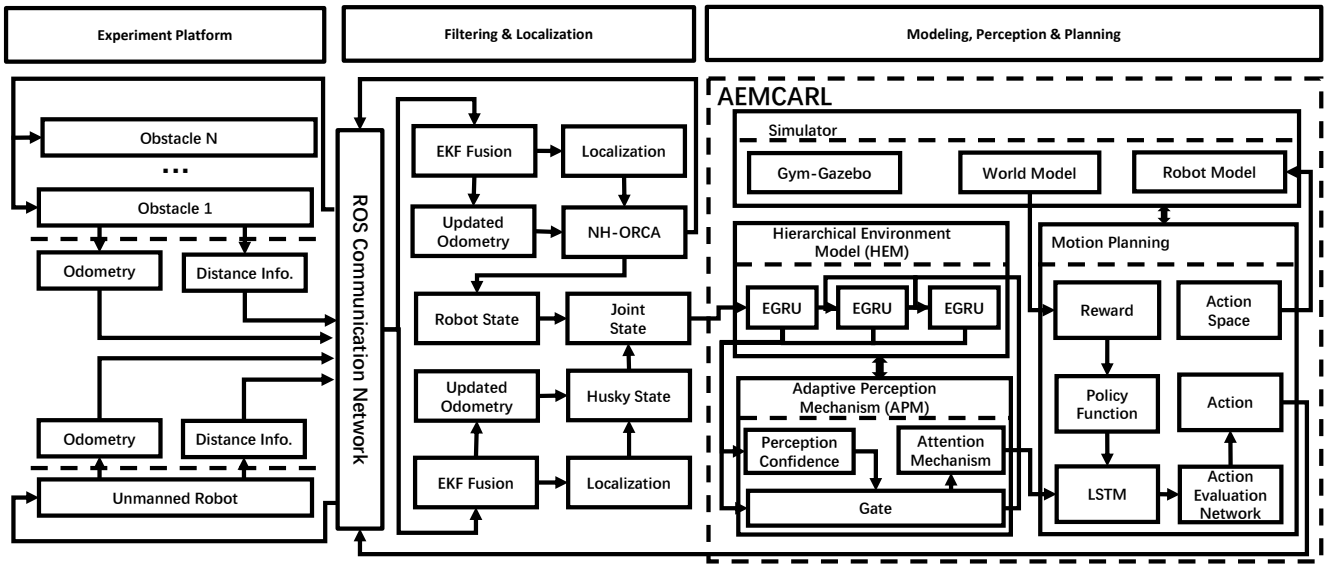

Fig. 2. The system diagram of the proposed crowd-aware RL-based collision-free navigation framework for an unmanned robot. The filtering and localization module is used to estimate the state information (pose and velocity) of mobile obstacles and the unmanned robot in the real world. These robotic mobile obstacles use NH-OCRA [18] for collision avoidance among themselves, keeping away from the unmanned robot if it is visible. The proposed framework, AEMCARL, includes a hierarchical environment model (HEM), an adaptive perception module (APM), and a motion planning module.

of the robot and obstacles at time t, $s_t = (r_t, o_t)$, is defined by Eq. (1).

$$\begin{aligned} r_t &= \left[v_x^r, v_y^r, x^r, y^r, g_x^r, g_y^r, v_r^{pref}\right], \\ o_t &= \left[v_x^o, v_y^o, x^o, y^o, v_o^{pref}\right]. \end{aligned} \tag{1}$$

The objective of the RL learning is to find the optimal motion planning policy, $\pi^* : s_t \mapsto a_t$, as shown in Eq. (2), which can enable the robot to reach the destination as soon as possible with low collision probability in crowded scenes.

$$\begin{aligned} \pi^*(\mathbf{s}_t) = &\underset{\mathbf{a}_t}{\operatorname{argmax}} R(\mathbf{s}_t, \mathbf{a}_t) + \\ &\gamma^{\Delta t \cdot v^{pref}} \int_{\mathbf{s}_{t+\Delta t}^{jn}} P(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+\Delta t}) V^*(\mathbf{s}_{t+\Delta t}) \, d\mathbf{s}_{t+\Delta t}, \end{aligned} \tag{2}$$

where a_t is the action taken by the robot at the time t, γ is the discount factor, $\gamma \in [0,1]$, and V^* is the optimal state-action value obtained by the trained value network.

Specifically, this work focuses on solving the following problems:

1) How to develop a model to represent the dynamic interactions among multiple agents with high accuracy and efficiency;
2) How to reduce the computational complexity of the system through the adaptive perception and the attention modules;
3) How to achieve the collision-free robust motion planning policy using a proper reward function and a filtering module.

IV. Proposed Methods

A. Hierarchical Environment Model

The action instantly taken by the unmanned robot is affected by both direct robot-to-agent interaction and potential agent-to-agent interaction. The goal of each agent in the real world is unpredictable and random. As shown in Fig. 3, the proposed HEM module consists of a number of extended GRUs (EGRUs) to represent the dynamic environment of multiple mobile obstacles. Each EGRU is given by

$$\begin{aligned} z_t &= \sigma\left(MLP([h_{t-1}, s_t], W_z)\right), \\ r_t &= \sigma\left(MLP([h_{t-1}, s_t], W_r)\right), \\ q_t &= \tanh\left(MLP([r_t \odot h_{t-1}, s_t], W_q)\right), \\ h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot q_t, \end{aligned} \tag{3}$$

where s_t is the input state defined in Eq. (7), each MLP consists of two linear perception layers, and all these MLPs have the same parameters. The output of each EGRU is given by

$$h_t^n = \begin{cases} EGRU(h_1, s_t^1), & \text{if } n = 1 \\ EGRU(h_t^{n-1}, s_t^n), & otherwise \end{cases} \tag{4}$$

where h_1 is initialized by a zero vector, n is the number of computation iterations of the AEM module, and t is the t^{th} environment input of the HEM module.

It is necessary to describe the group of obstacles as a whole. Therefore, an average operation is used to achieve embedding features of high dimension. Each feature contains both local and global information of obstacles.

B. Adaptive Perception Mechanism

The sigmoidal halting unit, $\sigma(\cdot)$, is added to the end of each EGRU, as shown in Fig. 3. It can be used to determine whether the output of the AEM module is sufficient to represent the dynamic environment or not through

$$p_t^n = \sigma\left(W^h h_t^n + b^h\right), \tag{5}$$

where W^h represents the weight parameters, and b^h is the bias parameter.

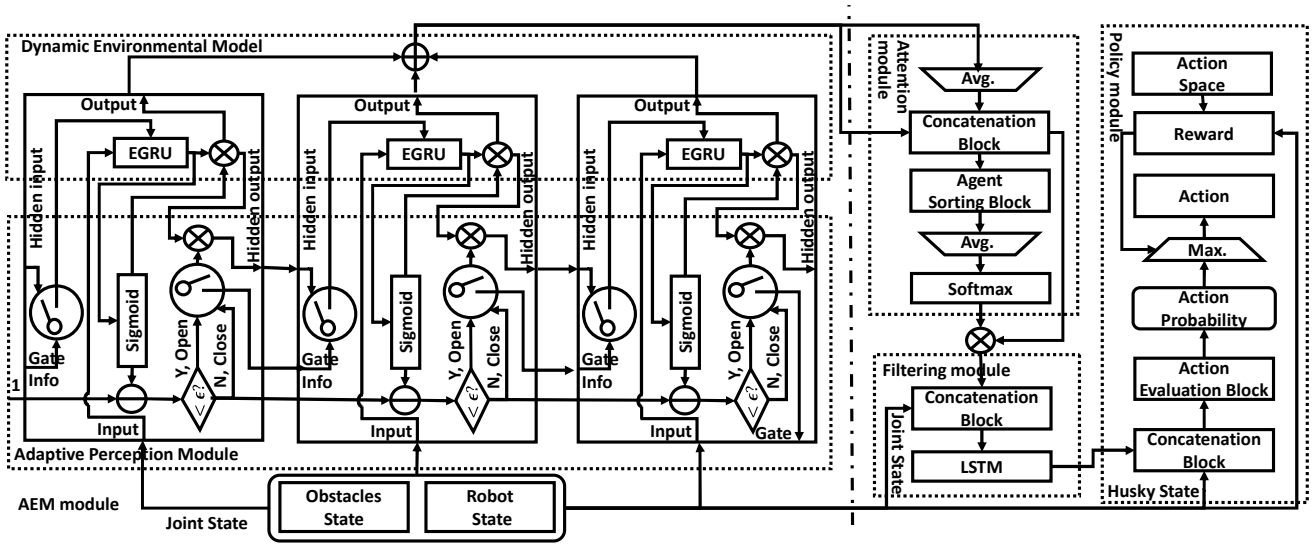

Fig. 3. The diagram of the adaptive environment model RL (AEMCARL) based crowded-aware collision-free motion planning network.

Algorithm 1 AEM Module

Initialization:

Input state = s_t, terminal state Nt = $\{h_t^n \geq 1-\varepsilon\}$, and initialize the hidden state h_1

Iteration:

for n = 1, N **do**
  Obtain the interaction features h_t^n with the EGRU
  Compute the believe probability p_t^n of h_t^n
  Update the sum of p_t^n and $p_t^n h_t^n$, respectively
  if terminal state N(t) or $t \geq t_{max}$ **then**
    Break
  end if
end for

Output:

Return the output feature y_t^n according to Eq. (6)

The output of the halt unit, given by (5), can also be used as the confidence of environment perception. The adaptive mechanism determines the number of EGRUs in use according to the perception confidence and the halt parameter, that is,

$$\begin{aligned} y_t^n &= \tfrac{1}{N(t)} \textstyle\sum_{n=1}^{N(t)} p_t^n h_t^n, \\ N(t) &= \min\{n : \textstyle\sum_{n=1}^{n} p_t^n \geq 1-\varepsilon\}, \ s.t.\ n \geq 1, \end{aligned} \tag{6}$$

where $N(t)$ is the adaptive number of EGRUs, and ε is the halt parameter, n is the maximum number of EGRUs, and the perception confidence is p_t^n. The output of the AEM module, which consists of an HEM module and an APM module, is y_t^n, given by Eq. (6). The overall operation of AEM module is shown in **Algorithm 1**.

The concatenation of the AEM output and its average, denoted by s_{IN}^{jn}, is used to represent the feature of multi-agent interactions, which can help the attention module to learn the global information of multiple agents. Then the input state, s_t, is concatenated with the embedding feature, s_{IN}^{jn}, to construct the final environment feature, s_{IN}, given by Eq. (7).

$$s_{IN} = s_{IN}^{jn} \bigoplus s_t, \tag{7}$$

where $\bigoplus$ means the concatenation operation.

An agent sorting block and a Softmax layer are used to construct the attention module (AM). The agent sorting block first assigns weights to obstacles according to their geometric and motion relationships to the robot in the environment. The Softmax layer gives attention scores based on the weights of obstacles. This procedure can help increase the speed of decision making and reduce the probability of collision. The final environment feature weighted by attention scores is denoted by s_{AM}.

C. Collision-free Policy Formulation

1) Reward Design: The reward is expected to be maximally returned by the optimal policy. In our reward function, a stationary penalty is added to avoid local optimum and boost the speed of the training process. Similar to the SARL/SARL-LM [4], [6], [9], the reward function is given by

$$R_t(\mathbf{s}_t, \mathbf{a}_t) = \begin{cases} -0.25 & \text{if } d_t < 0, \\ -0.1 + d_t/2 & \text{else if } d_t < 0.2, \\ -0.3 \times (d_s - d_f)/d_s & \text{else if } d_s < 0.2, \\ 1 & \text{else if } \mathbf{p}_t = \mathbf{p}_g, \\ 0 & \text{otherwise}, \end{cases} \tag{8}$$

where d_t is a safety distance used to keep the robot away from obstacles, d_s is a distance threshold to punish the robot for being around the freezing point, d_f is the predefined distance between robot position and the freezing point, p_t is the unmanned robot position at time t, and p_g is the goal position of the unmanned robot.

2) Policy Formulation: The LSTM layer can adaptively save the most important information of previous features in its memory unit; its recurrent structure can build a dynamic system to estimate more reliable features, despite

Algorithm 2 AEMCARL

Initialization:

Obtain the terminal state: {Reaching the goal, Collision}
Generate training state-action samples $\mathscr{S}$ using the ORCA method, and the memory unit $\hat{\mathscr{M}} \leftarrow \mathscr{S}$
Initialize the value network, $\mathscr{V} \leftarrow \mathscr{S}$, the target value network $\hat{\mathscr{V}} \leftarrow \mathscr{V}$

Iteration:

for epoch = 1, N **do**
  Initialize random samples s^0 from $\hat{\mathscr{M}}$
  repeat
    Formulate the interaction feature s_{IN} with the AEM
    Compute the concatenation state s_{AM} with the AM
    Update the value network $\mathscr{V}$ with the action module
    if terminal state s_t **then**
      Update the memory unit $\hat{\mathscr{M}}$
    end if
  until terminal state s_t or $t \geq t_{max}$
  Update the target network $\hat{\mathscr{V}} \leftarrow \mathscr{V}$
end for

Output:

Return $\mathscr{V}$

the input partial observations. Therefore, the features of environment, s_{AM}, are fed into the LSTM module to help achieve collision-free decisions for possible partial observations. The overall procedure of our proposed reinforcement learning of collision-free motion planning policy is shown in **Algorithm 2**.

V. Experiment Results and Discussions

A. Computation Setup

The simulation experiments are performed on a PC with an Intel core i7-7700K CPU, an Nvidia GTX1070 GPU, and 32G RAM. In the physical experiments, we use a portable computation platform with an Intel core i5-5500T CPU, and 16G RAM. The key parameters of the RL network include: discount factor $\gamma = 0.9$, batch size $batch = 100$, learning rate $L = 0.001$, stationary distance $d_f = 0.3$, and optimism method being Adam [19]. The hidden units of AEM module, attention module, and action module are [(100, 50)], [(100, 50), (50, 50, 1)], and [(100, 50, 1)], respectively. The action space consists of 80 discrete actions: 5 speeds exponentially distributed over $(0, v^{pref}]$ and 16 orientations evenly distributed over $[0, 2\pi)$.

B. Simulation Results

There are two testing cases for the unmanned robot navigation with multiple mobile obstacles: invisible and visible. The former means that all obstacles cannot detect the unmanned robot; whereas the latter means can. Fig. 4 illustrates the trajectories of a robot interacting with 5 mobile obstacles in the invisible case. Fig. 5(a) shows a simulation experiment with the Gym-Gazebo platform with 12 mobile obstacles.

Table II shows a comparison between the state-of-the-art methods and our method. The results of all other methods are

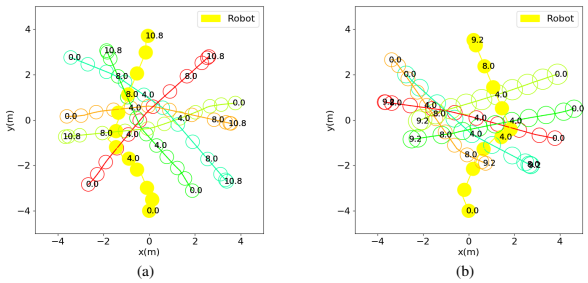

Fig. 4. Trajectories of the unmanned robot (yellow) and 5 mobile obstacles moving randomly. The maximum preferred velocity is 1m/s for all and the maximum radius of the obstacles is 0.2m. (a) The obstacles are all in the same size. (b) The obstacles are in random sizes.

TABLE II
A COMPARISON OF NAVIGATION PERFORMANCE BETWEEN THE STATE-OF-THE-ART METHODS AND OUR METHOD

Methods	Success (%)	Collision (%)	Time (s)
ORCA [8]	0.43	0.57	10.93
LM-SARL-Linear [6]	0.90	0.09	11.15
RGL-Linear [5]	0.92	0.04	10.35
MP-RGL-Onestep [5]	0.93	0.03	10.15
MP-RGL-MultiStep [5]	0.96	0.03	9.86
AEMCARL (Ours)	**0.98**	**0.02**	**9.23**

All experiments are based on one invisible unmanned robot and five mobile obstacles.

from literature [5]. The comparison is based on 5 obstacles for both invisible case with 500 random experiments. It can be seen that our method is the fastest one to reach the goal with the smallest collision probability.

Fig. 6 shows the comparison of system performance for different numbers of obstacles in the same size between two open-source baseline RL methods (LSTM-RL and SARL) and our method with 100 random experiments. Fig. 6(a) shows the success rate of SARL dramatically degrades with the increase of the number of obstacles; Fig. 6(a) and Fig. 6(b) show our method outperforms the two baseline methods in terms of both success rate and time to finish the task. Besides, since AEMCRL can always find a safe shortcut among crowds, minimum separation distances can be guaranteed despite the increased number of obstacles, as shown in Fig. 6(c).

C. Experiment Results

Our approach is also tested with real-world experiments in different scenarios using 4 to 8 mobile obstacles, respectively, as shown in Fig. 5(b). Turtlebots are used as mobile obstacles and a Husky is used as the unmanned robot that performs the RL-based collision-free navigation. All robots are equipped with UWB devices for distance measurements and localization. Fig. 7 shows the usage rate of three EGRUs for different scenarios. It can be seen that the proposed adaptive perception module (APM) can automatically change the structure of hierarchical environment module (HEM) in different scenarios to optimize the system computational complexity.

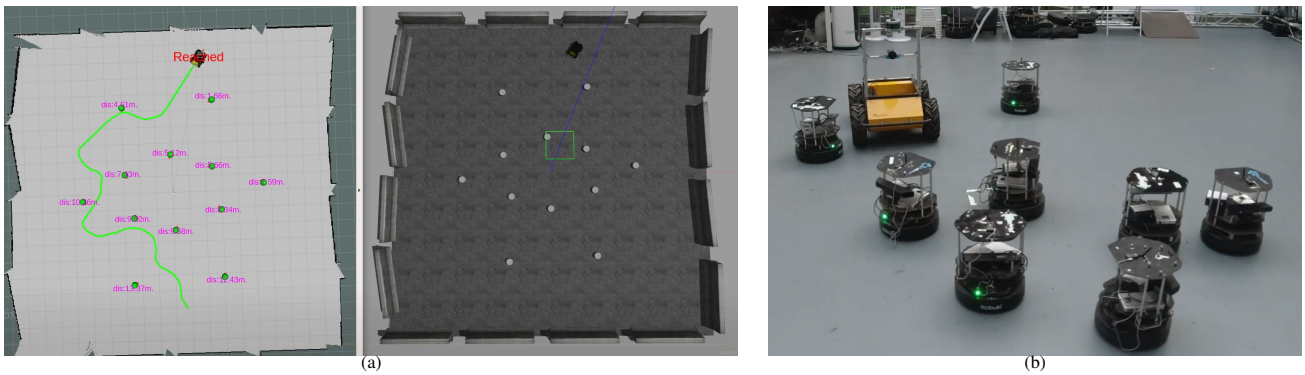

Fig. 5. An illustration of simulation and real-world experiments. (a) The Gym-Gazebo simulator: the green curve is the trajectory of the unmanned robot from the start position to the goal position. (b) The real-world experiment: one Husky as the unmanned robot and eight Turtlebots as the mobile obstacles.

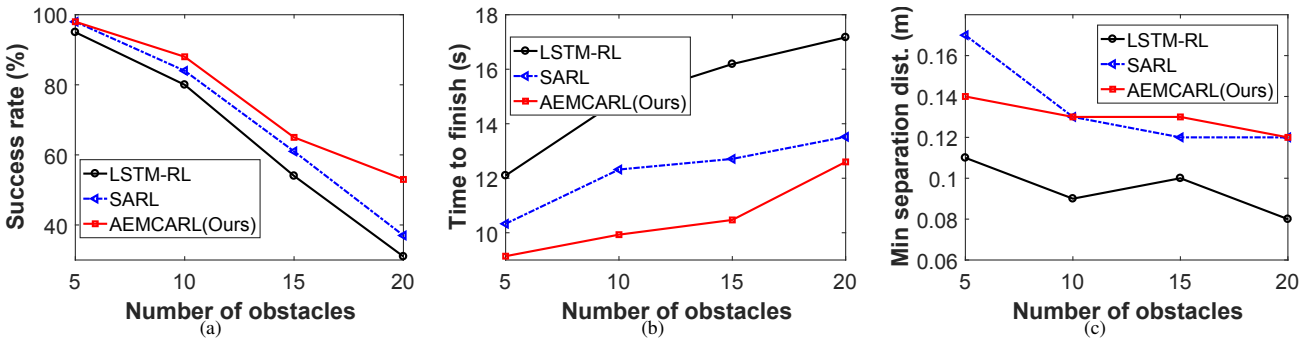

Fig. 6. The evaluation of AEMCARL. (a) A comparison of the navigation success rates using LSTM-RL, SAML, and AEMCARL for different numbers of obstacles. (b) A comparison of the time to finish the navigation task using LSTM-RL, SAML, and AEMCARL for different numbers of obstacle. (c) A comparison of minimum separation distances between obstacles and the unmanned robot using LSTM-RL, SAML, and AEMCARL for different numbers of obstacles.

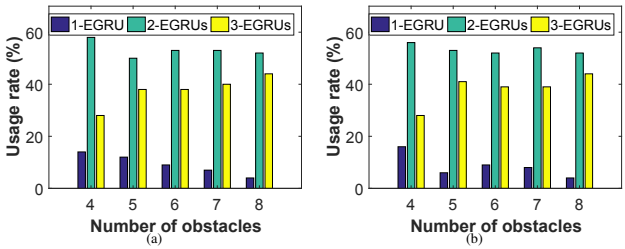

Fig. 7. An illustration of the usage rates of EGRUs in AEMCARL for different numbers of obstacles. (a) The unmanned robot is invisible to obstacles. (b) The unmanned robot is visible to obstacles.

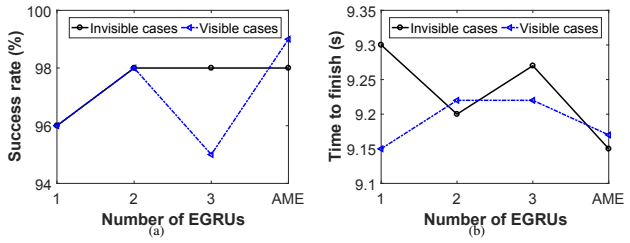

Fig. 8. (a) The success rates of the AEMCARL using different fixed numbers of EGRUs and the adaptive mechanism for 5 obstacles in invisible and visible cases, respectively. (b) The time to finish the navigation task of the AEMCARL using different fixed numbers of EGRUs and the adaptive mechanism for 5 obstacles in invisible and visible cases, respectively.

D. Discussions

Both SARL and AEMCARL use the attention mechanism to focus on the most threatening obstacles, which help them to achieve fast training convergence and best navigation performance in the case of 5 obstacles (Table II). By comparison, AEMCARL has fewer parameters (90% of SARL), and hence runs faster. The adaptive perception module can further help AEMCARL to improve the computation speed without losing the navigation performance. Fig. 8(a) and Fig. 8(b) show a comparison of navigation performance between using 3 different fixed numbers of EGRUs and the adaptive perception mechanism (AEM) given 5 obstacles for 50 random experiments of both visible and invisible cases. It can be seen that the AEM outperforms any fixed number of EGRUs.

VI. CONCLUSION

This paper has presented an adaptive environment model based reinforcement learning (AEMCARL) network which focuses on (1) dynamic environment representation, (2) adaptive perception mechanism, and (3) collision-free motion planning policy. The proposed hierarchical environment model (HEM) can robustly and efficiently represent the interactions among multiple agents. The adaptive perception mechanism (APM) is able to adaptively use computational resources within a certain degree of perception confidences. The policy decision module can achieve the real-time collision-free policy. The simulation and real-world ex-

periment results show that the proposed method outperforms state-of-the-art methods in terms of success rate and time to finish navigation tasks, especially for large numbers of mobile obstacles. Our future work includes extending the method to multi-robot collision-free navigation.

REFERENCES

[1] Y. F. Chen, M. Everett, M. Liu, and J. P. How, "Socially aware motion planning with deep reinforcement learning," *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017. [Online]. Available: http://dx.doi.org/10.1109/iros.2017.8202312

[2] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 797–803.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[4] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 961–971.

[5] C. Chen, S. Hu, P. Nikdel, G. Mori, and M. Savva, "Relational graph learning for crowd navigation," *arXiv preprint arXiv:1909.13165*, 2019.

[6] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," *arXiv preprint arXiv:1809.08835*, 2018.

[7] J. Van den Berg, M. Lin, and D. Manocha, "Reciprocal velocity obstacles for real-time multi-agent navigation," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1928–1935.

[8] J. Alonso-Mora, A. Breitenmoser, M. Rufli, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," in *Distributed autonomous robotic systems*. Springer, 2013, pp. 203–216.

[9] Y. F. Chen, M. Liu, M. Everett, and J. P. How, "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 285–292.

[10] P. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, *et al.*, "Interaction networks for learning about objects, relations and physics," in *Advances in neural information processing systems*, 2016, pp. 4502–4510.

[11] G. Ferrer, A. Garrell, and A. Sanfeliu, "Social-aware robot navigation in urban environments," in *2013 European Conference on Mobile Robots*. IEEE, 2013, pp. 331–336.

[12] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[13] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention in human crowds," in *2018 IEEE international Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4601–4607.

[14] A. Graves, "Adaptive computation time for recurrent neural networks," *arXiv preprint arXiv:1603.08983*, 2016.

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

[16] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[17] I. Sorokin, A. Seleznev, M. Pavlov, A. Fedorov, and A. Ignateva, "Deep attention recurrent q-network," *arXiv preprint arXiv:1512.01693*, 2015.

[18] S. C. Ruihua Han and Q. Hao, "A distributed range-only collision avoidance approach for low-cost large-scale multi-robot systems," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 8020–8026.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.