Deep reinforcement learning based socially aware mobile robot navigation framework

Nam Thang Do[1], Trung Dung Pham[1], Nguyen Huu Son[1], Trung Dung Ngo[2] and Xuan Tung Truong[1]

***Abstract*—In this study, we propose a socially aware navigation framework, which enables a mobile robot to avoid humans and social interactions in dynamic social environments, using deep reinforcement learning algorithm. The proposed framework is composed of two main stages. In the first stage, the socio-spatio-temporal characteristics of the humans including human states and social interactions are extracted and projected onto the 2D laser plane. In the second stage, these social dynamic features are then feed into a deep neural network, which is trained using the asynchronous advantage actor-critic (A3C) technique, safety rules and social constraints. The trained deep neural network is then used to generate the motion control command for the robot. To evaluate the proposed framework, we integrate it into a conventional robot navigation system, and verify it in a simulation environment. The simulation results illustrate that, the proposed socially aware navigation framework is able to drive the mobile robot to avoid humans and social interactions, and to generate socially acceptable behavior for the robot.**

2020 7th NAFOSTED Conference on Information and Computer Science (NICS) | 978-0-7381-0553-6/20/$31.00 ©2020 IEEE | DOI: 10.1109/NICS51282.2020.9335911

I. INTRODUCTION

To deploy mobile service robots in dynamic social environments, the robots should smoothly avoid humans and social interactions in their vicinity, while efficiency navigating towards a given goal. In order to achieve that, several socially aware robot navigation systems have been proposed in the recent years [1] and [2] to generate the socially acceptable behaviors for the mobile robot and ensure the human safety and comfort during the robot navigation.

The conventional socially aware robot navigation frameworks can be roughly classified into two categories according to the techniques utilized to develop the motion planning systems: (i) model-based methods and (ii) learning-based approaches. In the first category, the navigation systems utilize available models, such as artificial potential field [3], social force model [4], velocity obstacles [5] to develop the motion planning system. In the second category, the machine learning techniques, including reinforcement learning [6], and inverse reinforcement learning [7] techniques are used to enable the robots to navigate safely and socially in social environments.

Although the model-based approaches [8], [9], and [10] have been evaluated such that, the navigation systems are capable of driving the robot to navigate safely and socially towards a given goal, they still suffer essential weaknesses that seriously hinder the robot capabilities to navigate in crowded and dynamic environments. For example, in these papers, the authors have to hand-craft all the features of the human and surrounding environments, model these information and then incorporate them into the robot navigation system. In addition, several parameters are empirical set by the authors experiences for a specific environment. Moreover, this parameter set often need to be tuned individually, and can vary significantly for different environments.

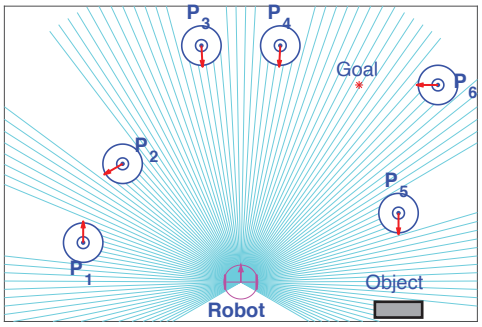


Fig. 1. An example of a dynamic social environment, in which there are a robot and six people $p_1 - p_6$. The people p_1 and p_2 are forming a stationary group, while a group of two people p_3 and p_4 are moving towards the robot, and person p_5 is interacting with an object. The robot needs to navigate to approach person p_6 while avoiding people $p_1 - p_5$, and social interactions.

To overcome the aforementioned drawbacks, a number of machine learning techniques-based navigation systems have been proposed to enable the robots to navigate socially and safely in dynamic social environments [11], [12], [13] and [14]. The learning-based approaches can be divided into two categories according to the information used as inputs of the the navigation system: (1) sensor-level-based approaches and (2) agent-level-based techniques. In the former, the authors utilize the raw data from sensors as input of the model. In the later, the agent information is utilized to develop the navigation system of the robot.

The sensor-level-based robot navigation system [11] and [12] learn to select actions directly from raw sensor readings (e.g. 2D laser rangefinder, rgb image, depth image) with end-to-end training. The raw sensor approach has the advantage that, the navigation system can deal with both static and dynamic obstacles (including walls). In addition, the data from the sensors can be fed directly into the neural network with a single framework. However in real-environments, especially in dynamic human environments, it is important to extract an agent-level representation of the world from multiple sensors.

Regarding to agent-level-based navigation system [13] and [14], the input of the model is the states including position and motion of nearby humans and moving obstacles. However, it is difficult to incorporate the surrounding environment

[1]Nam Thang Do, Trung Dung Pham, Nguyen Huu Son and Xuan Tung Truong are with Faculty of Control Engineering, Le Quy Don Technical University, Hanoi, Vietnam. xuantung.truong@gmail.com

[2]Trung Dung Ngo is with The More-Than-One Robotics Laboratory, University of Prince Edward Island, Canada dungnt@ieee.org

information into the model. In addition, since the number of surrounding humans can vary dramatically in different scenes, we need a model that can handle an arbitrary number of inputs into a fixed size output. Therefore, recently Everett et al. [13] uses a long short term memory network to process the state of each neighbor sequentially in reverse order of the distance to the robot. However, the underlying assumption that the closest neighbors have the strongest influence is not always true. Some other factors, such as motion and direction, are also essential for correctly estimating the importance of a neighbor, which reflects how this neighbor could potentially influence the robots goal acquisition.

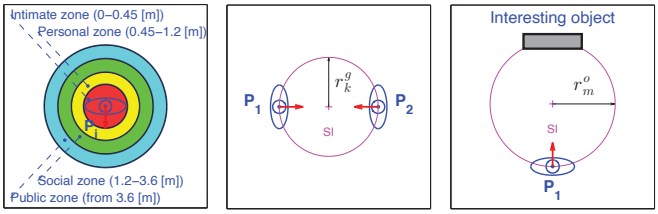
Although, learning-based navigation systems have been demonstrated that, the robot can navigate safely and socially in dynamic social environments. Each approaches have advantages and disadvantages. Therefore, to exploit the advantages of these techniques, in this paper we proposed a deep reinforcement learning-based socially aware navigation system, which can handle both agent-level and sensor-level. In addition, the proposed system takes into account the social interactions (human group and human–object interactions). In order to accomplish that, we project the human state and social interactions onto the 2D laser plane, and use this as input of a deep neural network. In addition, to exploit the parallel training process and help the robot to learn do deal with a variety of social situations in the dynamic social environments, we utilize the asynchronous advantage actor-critic (A3C) technique proposed by Mnih et al. [15] for teaching the robot. The main idea of the A3C algorithm is that, instead of using a single agent learning in a single environment, the algorithm uses multiple agents learning in a set of environments in parallel, and sharing knowledge with the others about what they have learned. This allows the algorithm to not only train faster as more agents are training in parallel, but also to increase the diversity of training data as each workerss experience is independent. As a result, the proposed framework is capable of generating the socially acceptable behavior for the robot and providing human safety and comfort in the robot's vicinity.

The rest of this paper is organized as follows. Section II presents the proposed socially aware robot navigation framework using the asynchronous advantage actor critic technique. Sections III describes the simulation results. We conclude our paper in section IV.

II. Proposed Framework

A. Problem Description

We consider a social context of a robot navigating towards a given goal (x_t^g, y_t^g) through a social environment in the presence of N humans and M objects, as shown in Fig. 1. Assume that, states of the robot in the robot local coordinate at time t are represented as $\mathbf{s}_t^r = [\dot{x}_t^r, \dot{y}_t^r]^T$, where $(\dot{x}_t^r, \dot{y}_t^r)$ is the robot's velocity. We also assume that, there are N people appearing in the robot's vicinity at time t, $P = \{\mathbf{p}_t^1, \mathbf{p}_t^2, ..., \mathbf{p}_t^N\}$, where $\mathbf{p}_t^i$ is the i^{th} person. The states of person $\mathbf{p}_t^i$ are represented as $\mathbf{s}_t^{pi} = [x_t^{pi}, y_t^{pi}, \theta_t^{pi}]^T$, where (x_t^{pi}, y_t^{pi}) is the position, θ_t^{pi} is the head orientation. There are M objects $O = \{\mathbf{o}_t^1, \mathbf{o}_t^2, ..., \mathbf{o}_t^M\}$, where $\mathbf{o}_t^j$ is the j^{th} interesting object. The states of object $\mathbf{o}_t^j$ are represented as $\mathbf{s}_t^{oj} = [x_t^{oj}, y_t^{oj}, \theta_t^{oj}]^T$, where (x_t^{oj}, y_t^{oj}) is the position, θ_t^{oj} is the orientation.



(a) (b) (c)

Fig. 2. The Hall model with four zones (a); The human group interaction with the radius r_k^g (b); The human–object interaction with the radius r_m^o (c).

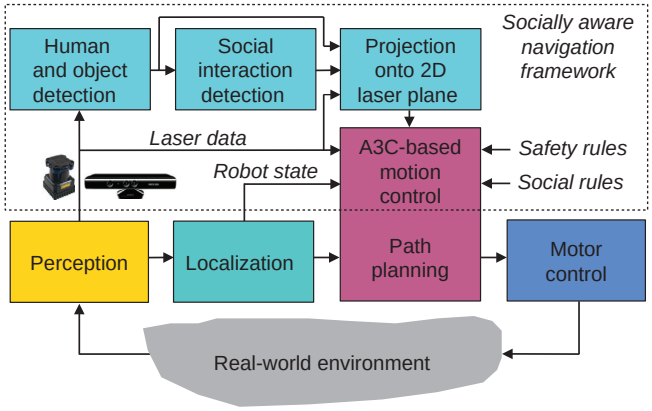


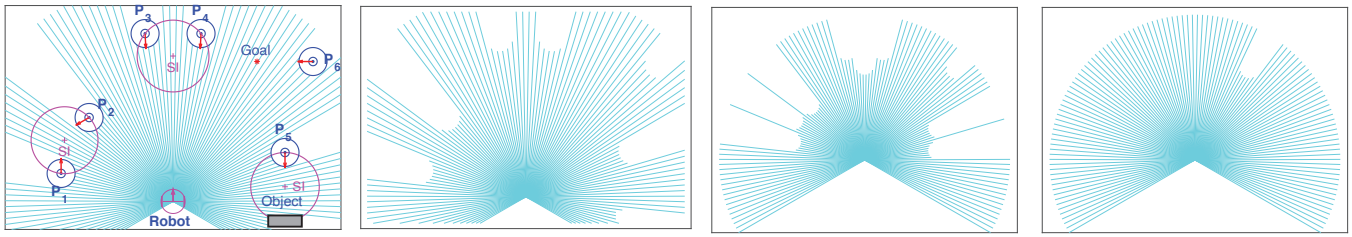
Fig. 3. Socially aware robot navigation framework based on asynchronous advantage actor-critic (A3C) learning technique.

To generate the socially acceptable robot behaviors, the robot should take the personal space around the human proposed by Hall [16], the human group interaction space (F-formation) proposed by Kendon [17], and the human–object interaction space proposed by Truong et al. [9], as shown in Fig. 2(a), 2(b) and 2(c), respectively.

B. Proposed Socially Aware Navigation Framework

As shown in Fig. 1, in this context, the robot should navigate towards the given goal timely while guaranteeing the safety and comfort of the humans in the robot's vicinity. In order to achieve that, the navigation system of the robot should take into account the obstacles, humans, social interactions. In addition, inspired by the idea of the A3C algorithm [15], instead of using a single robot learning to address social situations, we uses multiple robots learning in parallel to solve all social situations. To this end, we proposed a socially aware navigation framework for robot using the A3C algorithm and the conventional navigation system [18], as shown in Fig. 3.

In order to perceive the surrounding environment, the robot is equipped with a laser rangefinder and a Microsoft Kinect sensor, as shown in Fig. 3. The proposed framework first detects the humans and interesting objects using the data from



(a) Example scenario (b) Obstacle state space - $\mathbf{s}^o$ (c) Social state space - $\mathbf{s}^s$ (d) Goal state space - $\mathbf{s}^g$

Fig. 4. The state space of the mobile robot. (a) the scenario including humans, an object, a goal, and social interactions; (b) the state space of the obstacle avoidance layer; (c) the state space of the social layer; (d) the state space of the goal layer.

the laser and Kinect sensor. Then it recognizes social interactions in the robot's vicinity. The human and social interaction information are then projected onto 2D laser plane. Finally, the laser data, the projected human and social interaction, and the robot's state are used as inputs of the A3C-based motion control, which then generates the motion control command for the robot. This motion control command is then used to control the robot to approach the goal while avoiding other people, social interaction and obstacles in the robot's vicinity.

C. State Space

In this study, we utilize the 2D laser plane as state space of the robot. However, in order to incorporate the social dynamic of the humans (human states, social interactions) into the proposed socially aware navigation framework, we project these information onto the 2D laser plane, as shown in Fig. 4. To accomplish that, we assume the robot has 240 eyes pointing out in 240 directions, as seen in Fig. 4(a), in each direction the robot can observes 3 variables in its vicinity: (1) the distance from the robot to the surrounding objects, as shown in Fig. 4(b). We call this information as obstacle avoidance layer, and named as $\mathbf{s}^o$; the type of sensed objects including (2) humans and social interactions, as shown in Fig. 4(c). It is called social layer, and named as $\mathbf{s}^s$; and (3) goals, named as $\mathbf{s}^g$, as shown in Fig. 4(d). In addition, the robot's state is $\mathbf{s}_t^r = [\dot{x}_t^r, \dot{y}_t^r]^T$. Therefore, the total state space of the robot is $\mathbf{s}_t = [\mathbf{s}_t^o, \mathbf{s}_t^s, \mathbf{s}_t^g, \mathbf{s}_t^r]^T$. As a result, there is a total of $(240 \text{ x } 3)+2$ dimensional state space. In other words, the size of the state space is $N_s = 722$. This state space is then used as an input of a deep neural network.

D. Action Space

The action space of the robot is a set of permissible velocities. In this study, we utilize a differential drive robot platform, thus the action includes the linear and angular velocities, $a_t = [v_t, \omega_t]$. We consider the real robots kinematics and the real world applications, therefore we set the range of the linear velocity $v \in [0.0, 1.2]$ and the angular velocity $\omega \in [-\frac{\pi}{6}, \frac{\pi}{6}]$. We call A is the action space of the robot. Thus, at each time step the robot will select an optimal action in the action space $a_t \in A$. In this paper, we choose the size of the action space $N_a = 14$. Note that, the backward moving of the robot ($v < 0.0$) is not allowed since the laser rangefinder cannot cover the back area of the robot.

E. Network Architecture

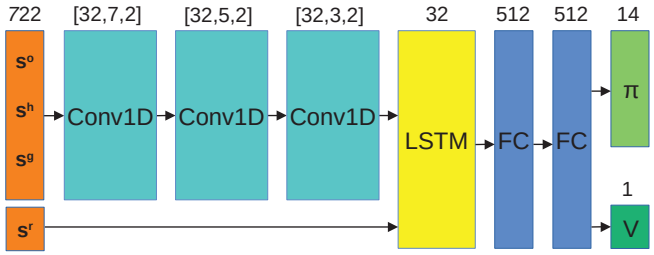


Fig. 5. The architecture of the policy and the value networks. The one-dimensional convolutional neural network (Conv1D) layer [32,7,2] indicates that, in this convolution layer the number of filters are 32, kernel size is 7 and stride is 2. The output of the final Conv1D is concatenated with the robot state, is then feed into the long short-term memory network (LSTM). The LSTM network has 32 hidden states. There are two fully connected (FC) layers of 512 neural nodes. The output of the policy network (π) is the action space of the robot. The output of the value network (V) is one value with linear activation function.

In this study, a deep neural networks-based function approximation is utilized to map from the state space to action space of the robot. Because, the neural networks offer many advantages, such as quality of the generalization, limited memory requirement for storing the knowledge, and continuous state space of the system. To deploy both the spatio-temporal information from the state space of the robot, we utilize the network architecture, as shown in Fig. 5.

Using this network architecture we aim at extracting the spatial features of the state space using three layers of one-dimensional convolutional neural network. We then feed these features into the LSTM network with 32 hidden state to extract the temporal dependencies of the state space. Following the LSTM network is two fully connected layers, which learn non-linear combinations of the features. The output of the value network (V) is used for learning process. The output of the policy network (π) is utilized to select the optimal action of the robot during the training and testing stages.

F. Reward Function

In order to guide the robot to learn how to avoid obstacles, humans and social interactions in the surrounding environment, and minimize the mean arrival time to the given target, we propose a reward function as follows:

$$r_t^r = r_t^{ro} + r_t^{rh} + r_t^{rs} + r_t^{rg} \quad (1)$$

where, r_t^r is the reward received by robot at time-step t. It is a sum of four components including r_t^{ro}–obstacle avoidance reward, r_t^{rh}–human avoidance award, r_t^{rs}–social interaction reward and r_t^{rg}–goal reward. Specifically, the robot is punished by r_t^{ro} when it collides with obstacles or other robots:

$$r_t^{ro} = \begin{cases} r_{collision}^{o} & \text{if } \sqrt{(x_t^o)^2 + (y_t^o)^2} < r^r + r^o \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

When the robot navigates close to a human or collides it, the robot is penalized by r_t^{rh} depending on the distance d_t^{rh} from the robot to the surrounding humans:

$$r_t^{rh} = \begin{cases} r_{collision}^{h} & \text{if } d_t^{rh} \leq r^r + r^p \\ r_{collision}^{ps} & \text{if } r^r + r^p < d_t^{rh} \leq r_0^{ps} \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where, $r_{collision}^{h}$ is the punishment when the robot collides with the humans; the robot is penalized by $r_{collision}^{ps}$ when it crosses the personal space of the humans; r_0^{ps} is the radius of the personal space according to Hall [16], as shown in Fig. 2(a).

When the robot crosses social interaction spaces including human group and human–object interaction spaces, it will be penalized by r_t^{rs} depending on the distance d_t^{rs} from the robot to the social interactions:

$$r_t^{rs} = \begin{cases} r_{collision}^{s} & \text{if } d_t^{rs} \leq r_k^g \text{ or } d_t^{rs} \leq r_m^o \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where, r_k^g and r_m^o are the radius of the human group and human–object interaction spaces, respectively.

In order to encourage the robot to quickly approach a given goal, and track the target when it is moving, the robot is awarded by r_t^{rg} for reaching and tracking the goal (x_t^g, y_t^g):

$$r_t^{rg} = r_t^{rag} + r_t^{rtg} \quad (5)$$

$$r_t^{rag} = \begin{cases} r_{arrival} & \text{if } \sqrt{(x_t^g)^2 + (y_t^g)^2} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

$$r_t^{rtg} = r_{tgoal} T_g \quad (7)$$

where, r_t^{rag} is awarded when the robot approaching the goal, and computed using Eq. 6; and when the robot maintain approaching goal in a interval of time T_g it is awarded r_t^{rtg}, which is calculated in Eq. 7.

G. A3C-based Socially Aware Navigation Framework

In this study, the A3C algorithm is used to train the robot learning how to navigate socially in dynamic social environments. To accomplish that, we use multiple robots learning in parallel with each robot having its own network parameters and a copy of the environment. The robots will interact with their respective environments asynchronously, and learning with each interaction. Each robot is controlled by a global network. As each robot gains more knowledge, it contributes to the total knowledge of the global network. The presence of a global network allows each robot to have more diversified training data.

Algorithm 1: A3C-based socially aware mobile robot navigation algorithm

input : State size N_s, action size of the robot N_a, number of robot N_r, maximum global shared episode T_{max}, learning rate α.
output: The trained global shared policy network
1 **begin**
2 Initialize global shared policy network using N_s, N_a
3 Initialize global shared value network using N_s
4 Initialize N_r pair of actor-critics with local policy and value networks
5 Assign each actor-critic to a thread
6 Start N_r threads
7 **repeat**
8 Asynchronous update the global shared networks using local gradients from each actor-critic
9 **until** all threads are terminated

The A3C-based socially aware navigation framework of the robot is presented in Algorithms 1. The inputs of the algorithm are the size of the state space N_s and the action size N_a, the maximum global shared episode T_{max}, and the learning rate α. While the output is the trained global shared policy network, which then is used to choose the optimal action of the robot. We first initialize the global shared policy and value networks using the network architecture presented in Section II-E (Lines 2 and 3 of the Algorithms 1). We also initialize N_r pair of actor-critics (Line 4 of the Algorithms 1), which is corresponding to N_r robots. Each robot is assigned local policy and value networks, which their architecture are similar to the global networks. We then assign each robot to a thread, and start the threads (Lines 5 and 6 of the Algorithms 1). The global shared networks are asynchronous updated the using local gradients from each robot.

Algorithm 2 shows the learning process of each robot (each thread). The inputs of the algorithm are the state space $\mathbf{s}_t$, action space A, maximum global shared episode T_{max}, maximum step of each episode t_{max}, frequency update t_{update} and discount factor γ. Whereas, the outputs are the accumulate gradients of the policy network $d\theta$ and the value network $d\theta_v$, which are used to asynchronous update the global shared networks. At the first of each episode, the local networks will be updated with new weights from the global shared networks (Line 4 of the Algorithm 2), and the robot will get the first state (Line 5 of the Algorithm 2). The algorithm will loop until t_{max} time step or the robot collides with humans, obstacles (Line 6 of the Algorithm 2). In each loop, the robot first selects the action according to the local policy network, then performs this action, and receive new state and reward (Lines 7 and 8 of the Algorithm 2). At each t_{update} or the robot collides with the environment, the algorithm will calculate the accumulate policy and value gradients, which are then used to asynchronous update the weights for the global networks (Lines 11–16 of the Algorithm 2).

Algorithm 2: Actor-critic thread in A3C algorithm

input : State space $\mathbf{s}_t$, action space A, maximum global shared episode T_{max}, maximum step of each episode t_{max}, frequency update t_{update}, discount factor γ.
output: Accumulate gradients $d\theta$ and $d\theta_v$

begin
  Initialize memory M to capacity t_{update}
  for $episode = 1, T_{max}$ **do**
    Update local networks using global shared networks $\theta' \leftarrow \theta$ and $\theta'_v \leftarrow \theta_v$
    Get current state $\mathbf{s}_t$
    while $((t < t_{max})$ **or** $(not\ terminal))$ **do**
      Select a_t according to policy $\pi(a_t|\mathbf{s}_t;\theta')$
      Perform action a_t, receive state $\mathbf{s}_{t+1}$, reward r_t
      Store transition $(\mathbf{s}_t, a_t, r_t, \mathbf{s}_{t+1})$ in memory M
      if $(t$ **mod** $t_{update} == 0)$ **or** $(terminal)$ **then**
      // Accumulate gradients $d\theta$, $d\theta_v$
        for $i = 1, length(M)$ **do**
          $R \leftarrow r_i + \gamma R$
          $A = R - V(\mathbf{s}_i;\theta'_v)$
          $d\theta \leftarrow d\theta + \nabla_{\theta'} log\pi(a_i|\mathbf{s}_i;\theta')A$
          $d\theta_v \leftarrow d\theta_v + \partial(R - V(\mathbf{s}_i;\theta'_v))^2/\partial\theta'_v$
        end for
        Asynchronous update θ, θ_v using $d\theta$, $d\theta_v$
        Clear the memory M
      end if
      $\mathbf{s}_t \leftarrow \mathbf{s}_{t+1}$; t $\leftarrow$ t +1
    end while
  end for

H. Training

1) Training Environment: In order to guide the robot learn to avoid humans, social interactions, obstacles, and approach goals, we have created a hallway-like scenario with walls, interesting objects, humans, social interactions, and goals based on the Stage robot simulator [19], as depicted in Fig. 6. The moving humans, goals and objects are controled using social force model proposed by Helbing et al. [4] and the available software platform[1].

2) Training Setup: The software core of the robot is based on the Robot Operating System (ROS) [20]. The proposed framework is implemented using Python and the Tensorflow library[2]. We adopt the Adam optimization method [21] for training the proposed model. We train the proposed socially aware navigation framework on an Intel core i7 4.2 GHz CPU desktop. The running time is 16 hours with about 1000 global episodes. The parameters of the training and testing process are empirically set in Table I.

3) Training Process: In order to help the robot easy to learn, we divide the training process into three stages including

[1]http://pedsim.silmaril.org
[2]https://www.tensorflow.org

TABLE I
PARAMETERS SET IN TRAINING AND TESTING PROCESS

Parameter	Value	Parameter	Value	Parameter	Value
α	0.001	γ	0.99	t_{update}	20
N_s	722	N_a	14	r_0^{ps}	0.9 [m]
t_{max}	500	$r^o_{collision}$	-5	$r^h_{collision}$	-10
$r^{ps}_{collision}$	-0.2	$r^s_{collision}$	-0.2	$r^a_{arrival}$	10
r_{tgoal}	0.1	N_r	8		

(i) static stage, (ii) semi-dynamic stage and (iii) dynamic stage. In the static stage, all of the humans and goals in the scenario are stationary. In the second stage, all goals in the scenario are moving. In the final stage, the moving humans and the goals are navigated around the scenario.

The output of the A3C-based motion control system is the optimal control command of the robot $a_t = [v_t, \omega_t]$. It is then used to drive the differential drive robot platform to socially navigate in dynamic social environments, providing the safety and comfort for the human and socially acceptable behavior for the robot.

III. Simulation Results

To validate the proposed socially aware navigation framework, we adopt the *social individual index* (SII) proposed by Truong et al. [22]. The SII value is used to estimate the human safety and comfort, and socially acceptable behaviours of the mobile robot.

The mobile robot is requested to navigate around the scenario while avoiding the humans and social interactions in its vicinity. The simulation results are shown in Fig. 7. As can be seen in Fig. 7, the SII value is smaller than 0.14 (the blue line). It indicates that the robot often maintains a comfort distance to the humans.

In summary, the simulation results clearly indicate that the proposed socially motion model is capable avoiding humans and human groups in socially acceptable manners while still guaranteeing the human safety and comfort in crowded and dynamic social environments.

IV. Conclusions

We have presented a socially aware navigation framework, which enables a mobile robot to avoid humans and social interactions in dynamic social environments, using deep reinforcement learning algorithm. The proposed framework is composed of two main stages. In the first stage, the socio-spatio-temporal characteristics of the humans including human states and social interactions are extracted and projected onto the 2D laser plane. In the second stage, these social dynamic features are then feed into a deep neural network, which is trained using the asynchronous advantage actor-critic (A3C) technique, safety rules and social constraints. The trained deep neural network is then used to generate the motion control command for the robot. We incorporate the proposed motion model into a conventional robot navigation system, and verify it in a simulation and environment. The experimental results show that, the proposed socially aware navigation

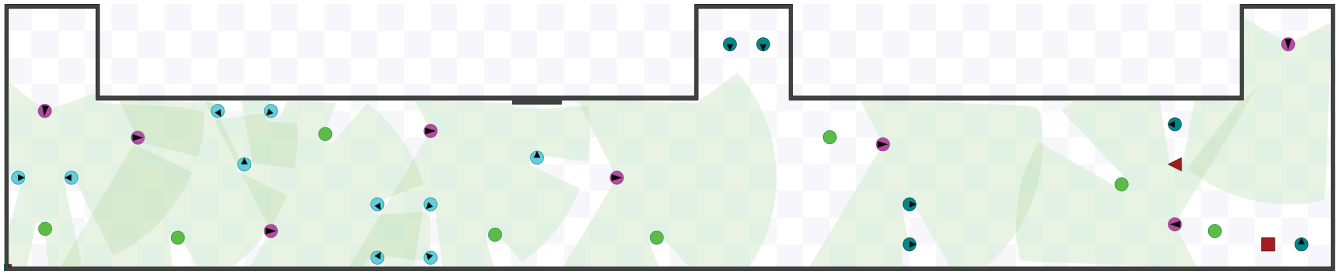


Fig. 6. A hallway-like scenario with walls, interesting objects, humans, and goals. Eight robots (magenta dots), 8 typical social situations, 10 stationary people (cyan dots), 6 moving people (dark blue dots), and two moving object (brown triangle and square), and 8 goals (green dots) are distributed in the scenario. The robot is assigned a task to navigate to approach goals while avoiding humans and social situations: (1) a group of two standing people; (2) a group of three standing people; (3) a group of four standing people; (4) a human–object interaction; (5) a group of two people moving cross the scenario; (6) a group of two people moving forward; (7) a human–object moving forward; (8) a human–object moving cross the scenario.

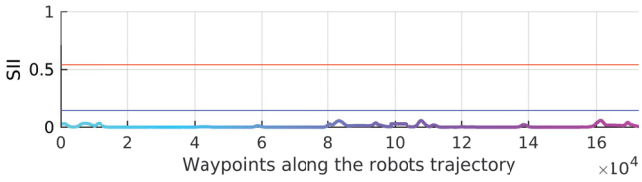


Fig. 7. The social individual index (SII) value.

framework enables the mobile robot to avoid humans and social interactions, providing socially acceptable behavior for the robot.

In the future, we will evaluate performance of the proposed framework in several social situations, especially in real-world environments. In addition, we will extend our proposed framework to three dimensional state space by using the RGB-D image sequence as the input of the deep neural network.

ACKNOWLEDGMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under grant number 102.01-2018.10.

REFERENCES

[1] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch, "Human-aware robot navigation: A survey," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1726–1743, 2013.

[2] J. Rios-Martinez, A. Spalanzani, and C. Laugier, "From proxemics theory to socially-aware navigation: A survey," *International Journal of Social Robotics*, vol. 7, pp. 137–153, September 2014.

[3] O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 500–505, March 1985.

[4] D. Helbing and P. Molnr, "Social force model for pedestrian dynamics," *Physical Review E*, pp. 4282–4286, 1995.

[5] P. Fiorini and Z. Shillert, "Motion planning in dynamic environments using velocity obstacles," *International Journal of Robotics Research*, vol. 17, pp. 760–772, 1998.

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

[7] K. Henrik, S. Markus, S. Christoph, and B. Wolfram, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1289–1307, 2016.

[8] M. Shiomi, F. Zanlungo, K. Hayashi, and T. Kanda, "Towards a socially acceptable collision avoidance for a mobile robot navigating among pedestrians using a pedestrian model," *International Journal of Social Robotics*, vol. 6, no. 3, pp. 443–455, 2014.

[9] X. T. Truong and T. D. Ngo, "Dynamic social zone based mobile robot navigation for human comfortable safety in social environments," *International Journal of Social Robotics*, vol. 8, no. 5, pp. 663–684, 2016.

[10] D. Claes and K. Tuyls, "Multi robot collision avoidance in a shared workspace," *Autonomous Robots*, vol. 42, pp. 1749–1770, December 2018.

[11] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 31–36, September 2017.

[12] T. Fan, P. Long, W. Liu, and J. Pan, "Fully distributed multi-robot collision avoidance via deep reinforcement learning for safe and efficient navigation in complex scenarios," *arXiv preprint arXiv:1808.03841*, 2018.

[13] M. Everett, Y. F. Chen, and J. P. How, "Motion planning among dynamic, decision-making agents with deep reinforcement learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3052–3059, October 2018.

[14] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," *arXiv:1809.08835v2*, 2018.

[15] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Harley, T. P. Lillicrap, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 1928–1937, 2016.

[16] E. T. Hall, *The hidden dimension: man's use of space in public and private*. The Bodley Head Ltd, London, 1966.

[17] A. Kendon, *Conducting interaction: Patterns of behavior in focused encounters*. Cambridge University Press, 1990.

[18] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots*. The MIT Press, February 2011.

[19] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The player/stage project: Tools for multi-robot and distributed sensor systems," in *In Proceedings of the 11th International Conference on Advanced Robotics*, pp. 317–323, 2003.

[20] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Ng, "ROS: an open-source Robot Operating System," in *ICRA Workshop on Open Source Software*, vol. 32, pp. 151–170, 2009.

[21] D. P. Kingma and J. B. Adam, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations*, pp. 1–15, 2015.

[22] X. T. Truong and T. D. Ngo, "To Approach Humans?: a unified framework for approaching pose prediction and socially aware robot navigation," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 10, no. 3, pp. 557–572, 2017.