Autonomous Navigation for Omnidirectional Robot Based on Deep Reinforcement Learning

Van Nguyen Thi Thanh
Hanoi Vocational College of High Technology, Hanoi, Vietnam
Email: vanntt@hht.edu.vn

Tien Ngo Manh, Cuong Nguyen Manh, Dung Pham Tien, and Manh Tran Van
Institute of Physics, Vietnam Academy of Science and Technology, Hanoi, Vietnam
Email: {tieniop, manhcuong313.ng, phamdung210597, manhtran4321}@gmail.com

Duyen Ha Thi Kim and Duy Nguyen Duc
Hanoi University of Industry, Hanoi, Vietnam
Email: ha.duyen@haui.edu.vn, duyndfx01662@funix.edu.vn

***Abstract*—This paper presents navigation for four-wheel Omni Robot using architecture deep reinforcement learning in an unknown environment. A deep reinforcement learning algorithm combines with effectively training data using stochastic gradient updates in order to connect a goal. The approach is simulated and visualized using Gazebo and is updated via policies trained by deep Q learning network. Using recent deep-learning techniques as the basis of the framework, our results indicate that it is capable of providing smooth navigation for the Omni robot in exploring unpredicted surroundings. Once extended to real-world operation, this framework could enable the Omni Robot to gain achievement for self-driving tasks.**

***Index Terms*—deep reinforcement learning, omni robot, Robot operating system, navigation**

I. Introduction

In recent years, a significant number of practical applications for Omni robots have been received much attention in the scientific community, especially applications in relation to navigation for autonomous robots. More importantly, the deep learning methods are enable Omni robots to interact and navigate environments as human interaction ways. For example, by gathering data during the interaction and using end-to-end reinforcement learning [1], robots are able to learn interaction behavior through high dimensional sensory and camera information. Clearly, the visual sensor system is one of the keys to solving any tasks in relation to navigation, path detection for robots in unpredicted situations. Besides, in order tackle problems of obstacle avoidance, reinforcement learning, which plays differently from supervised and unsupervised learning branches, is becoming an efficient solution for autonomous robots nowadays [2], [3].

Manuscript received December 15, 2019; revised March 9, 2020.

With the purpose of putting machines close to the human perception of artificially intelligent, the reinforcement learning approach is developed to address the problem of agent learning to operate in an environment through maximizing a scalar reward signal. Over the past decades, there are numerous publications that concentrate on the improvement of deep reinforcement learning (DRL). One of these is the successful game Atari 2600 or Go games in [4], [5] by observing just the screen pixels and receiving a reward calculated taking into consideration the game score. Reinforcement learning was successfully combined with a convolutional neural net to approximate the action value. Other successful works have shown in [6], Arun Nair at el displays architectures of deep reinforcement learning (DQN) algorithms by optimizing approximately the Bellman equation. In terms of an autonomous robot, some papers have been published in [7]-[9] shown that the Q learning network is trained by sampling mini-batches of experiences from buffer uniformly at random.

However, applying particularly reinforcement learning in robot setting suffers from many challenges since the high dimensionality and continuous states and actions of the robot [10]. In a simulation, creating an accurate model robot and its environment are challenging and often require a lot of sufficient data samples. To address these dilemmas, the operation of learning of robots is simulated and designed in Gazebo since its compatibility with the complex structure of the robot. More than that, Gazebo enables to construct of a virtual environment [11], which is imperative in the process of scrutinizing reinforcement learning algorithms. Beside, one of major problems in mobile robot navigation is that how the robot can find a collision free path from its starting point. With reinforcement learning algorithms integrated in Gazebo, several methods have been proposed to deal with obstacle avoidances [12]-[15].

This paper aims to illustrate how the Omni robot performs navigation using model-free deep Q learning to

doi: 10.18178/ijmerr.9.8.1134-1139

navigate in unpredicted environments. It will also explain how to obtain the policy when such a model is unknown in advance by using a virtual environment to conduct in simulation. In Section II, the background of deep reinforcement learning is presented in part A, then the deep Q learning algorithm for Omni robot has been illustrated. After that, the architecture of Omni robot with Q network is shown in Section III. Research results in 3D simulation using Gazebo is included in Section IV. Finally, a conclusion is mentioned in Section V.

II. Deep Reinforcement Learning

A. Background

Markov processes and Markov decision processes are widely used in computer science and other engineering fields. In reinforcement learning, Markov decision process (MDP) is applied to calculating sequence S_t as the state representation at time t. In other words, the Markov property requires the states of the system to be distinguishable from each other and unique.

A typical MDP is represented using a 6-tuple (S, A, T , γ , D, R), where S is a (finite) set of possible states that represent a dynamic environment, A is a (finite) set of available actions that the agent can select at a certain state,1 T is the state transition probability matrix that provides the probability of the system transition between every pair of the states, γ ∈ [0, 1) is the discount rate that guarantees the convergence of total returns, D is the initial-state distribution, and R is the reward function that specifies the reward gained at a specific state by taking a certain action.

$$\mathrm{P}(\mathrm{s}_{t+1}|\mathrm{s}_t,\mathrm{a}_t,\mathrm{s}_{t-1},\mathrm{a}_{t-1},...,\mathrm{s}_0,\mathrm{a}_0)=\mathrm{P}(\mathrm{s}_{t+1}|\mathrm{s}_t,\mathrm{a}_t) \quad (1)$$

$$\pi^* = \arg\max \mathrm{E}\left[\sum_{t=0}^{\infty}\gamma^t R(\mathrm{s}_t,\pi(\mathrm{s}_t))\right]\infty \quad (2)$$

Reward the agent receives by taking an action determined by policy π at the present state. Given a policy π, the MDP in (1) is reduced to a Markov chain with transition probabilities, given by where γ is the discount rate.

A Q-network can be trained by minimizing a sequence of loss functions that changes at each iteration i .

$$L_i(\theta_i) = E_{s,a\sim p(\cdot)}\left[(Q^{target} - Q(s,a;\theta_i))^2\right] \quad (3)$$

where $Q^{target} = E_{s'\sim\varepsilon}\left[r+\gamma\max_{a'}Q(s',a';\theta_{i-1})\middle|s,a\right]$ is the target for iteration I and $p(\mathrm{s},\mathrm{a})$ is a probability distribution over sequence s and action a.

The parameters θ of the so-called Q-network are optimized so as to approximately solve the Bellman equation. Differentiating the loss function with respect to the weights we arrive at the following gradient:

$$\nabla_{\theta_i}L_i(\theta_i) = E_{s,a\sim p(\cdot);s'\sim\varepsilon}\left[\begin{pmatrix}\mathrm{r}+\gamma\max\mathrm{Q}^*(\mathrm{s}',\mathrm{a}';\theta_i)\\ -\mathrm{Q}(\mathrm{s},\mathrm{a};\theta_i)\end{pmatrix}\nabla_{\theta_i}Q(\mathrm{s},\mathrm{a},\theta_i)\right] \quad (4)$$

The Agent's behavior at each time-step is selected by an ε-greedy policy where the greedy strategy is adopted with probability 1 – ε while the random strategy with probability ε.

B. Learning Procedure for Deep Q Learning of Omni Robot in ROS

The system observes the current scene including depth frames, and take action based ε-greedy strategy. The interaction experience $e=(\mathrm{s}_i,\mathrm{a}_i,\mathrm{r}_i,\mathrm{s}_{i+1})$ is stored in replace memory $\mathbb{M}$ keeps N most recent experiences by interacting with the environment , then DQN agent samples the mini batch from replay memory and train network on this mini batch.

Algorithm for Deep Q learning with replace memory

Initialize replay memory $\mathbb{M}$ to size N
Initialize the Q- network $Q(\mathrm{s},\mathrm{a},\theta)$
Initialize the Omni robot system
For episodes = 1, $\mathbb{M}$ do
  Initialize sequence s1 = {x1} and sequence $\phi=\phi(\mathrm{s}_1)$
  For t, 1, T do
    With probability ε select a random action at a_t
    Otherwise select $\mathrm{a}_t=\max Q^*(\phi(s_t),a;\theta_i)$
    $s_{t+1},r_t \leftarrow$ Execute action a_t observe reward r_t
    Store transition $(\phi_t,\mathrm{a}_t,\mathrm{r}_t,\phi_{t+1})$ in $\mathbb{M}$
    Sample random mini batch of $(\phi_t,\mathrm{a}_t,\mathrm{r}_t,\phi_{t+1})$ from $\mathbb{M}$
    Set:
    $$Q^* = \begin{cases} r_i & \text{for terminal } \phi_{i+1} \\ r_i+\gamma\max_{a'}Q(\phi_{j+1},\mathrm{a}';\theta) & \text{for non-terminal } \phi_{i+1}\end{cases}$$
    Perform a gradient descent step on $(y_i - Q(s,a;\theta_i))^2$ according to equation (4)
  End for
End for

III. The Architecture of Control System of Omni Robot

A. Robot Controller Design

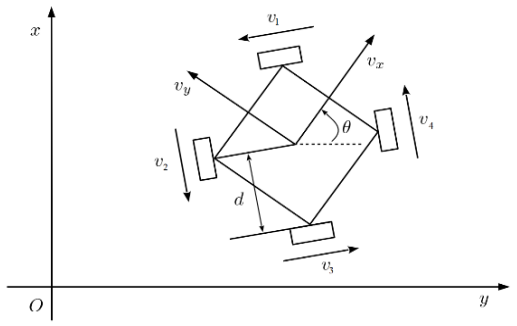

Figure 1. Omni robot.

The Omni robot has four wheels which are 90 degrees apart. The robot movement would be identified for the

navigation stack. As the global coordinate chosen in Fig. 1, it is obvious that the velocity of the robot contains three components: a linear velocity along Ox-axis and Oy-axis an angular velocity.

The robot's coordinate vector is defined as $\underline{q} = \begin{bmatrix} x & y & \theta \end{bmatrix}^T$ and the velocity of the robot in the global coordinate could be obtained by taking the derivative of $\underline{q}$. Oxy represents the global coordinate axis, the distance between wheels and the robot center was defined by d .To facilitate the robot controller design, the relationship between the velocity in the robot's axis and the velocity in the world's axis is described by the kinematic model of the robot:

$$\dot{\underline{q}} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \underline{v}. \tag{5}$$

From (1), we obtain the equations which would be used to program the robot's navigation in ROS:

$$\begin{cases} \dot{x} = v_x \cos\theta - v_y \sin\theta \\ \dot{y} = v_x \sin\theta + v_y \cos\theta \\ \dot{\theta} = w \end{cases} \tag{6}$$

where v_x, v_y, and $\dot{w}$ are the velocity in the robot's axis and the velocities is the control signal generated from the local planner. The robot position is directly controlled by these signals which are transformed into the desired signals for four wheels of the robot. The transformation formula is described as below:

$$\begin{cases} v_x = \dfrac{\sqrt{2}}{8}(-v_1 - v_2 + v_3 + v_4) \\ v_y = \dfrac{\sqrt{2}}{8}(v_1 - v_2 - v_3 + v_4) \\ \omega = \dfrac{(v_1 + v_2 + v_3 + v_4)}{4d} = r\dfrac{(\omega_1 + \omega_2 + \omega_3 + \omega_4)}{4d} \end{cases} \tag{7}$$

With the references value transformed from the local planner, the velocity for each wheeled can be computed to ensure the robot to track the desired value

B. Architecture Omni with Deep Learning

To handle the large number of state-action pairs in autonomous mobile robot avoiding obstacle problems, we implemented deep Q-learning using a neural network to select appropriate action with environment. Fig. 2 shows that the structure of the robot learning system with a Q network.

The Omni robot selects 8 actions shown as Fig. 3 by Q-values and reflects it in the environment by observation spaces. Afterward, the reward will be stored from an environment in replace memory, which is inputs of Omni as sample random transitions for training the Q-network.

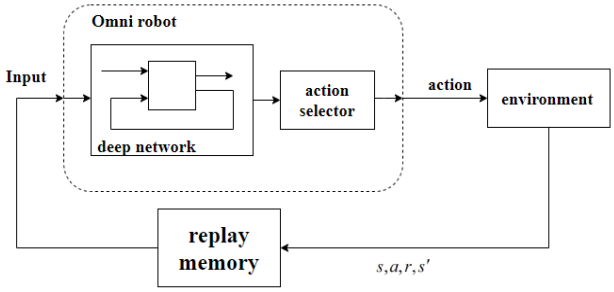

Figure 2. Architecture of robot learning system based on deep network.

The input of the network is state vectors generated by concatenating 26 range-scan measurements with the angle and distance to the objective. In the range of 360 degree of Lidar, robot scan any obstacles with its max radius 20 m, the output of eight possible actions is interpreted as movement of a straight line. the CNN network comprises of three convolution layers and two fully connected layers with a single linear output. When using simple CNN network, Omni robot is capable of using graphical computing units for faster.

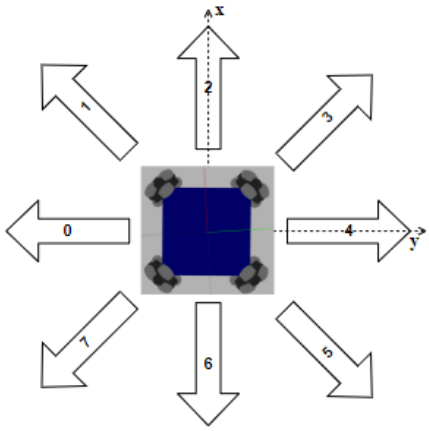

Figure 3. Actions of Omni robot.

During training, we used the RMSPropOptimizer because of its lowest variance. The network parameter is installed by the learning rate of 0.005, decay of 0.99. The learning rate was decreased during the training as this also proved to stabilize the learning.

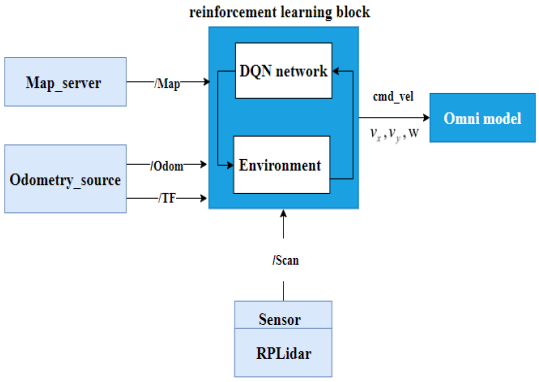

Figure 4. Architecture of robot learning system in Gazebo.

In order to overcome the dimensional problems of robot, the Omni robot system is installed in Fig. 4 including main blocks, namely map_server, Odometry, Robot controller, and Reinforcement learning, which are encapsulated in nodes. In Gazebo, sensors like IMU and RPLidar are the importance components to set up information states of Omni robot. The laser sensor is used to recognize the obstacles around the robot in the virtual map while the current position of Omni robot is updated by using the odometry mearsurements from Odometry_source in environment. Map_server is used for updating the map with the new obstacle. During trial-and-error with the unknown environment, Reinforcement learning block returns angular and linear velocity. DQN network is also updated after 50 epochs in order to modify appropriate weights for network that estimate what the optimal Q-value by defining clearly it interpretation between action and states of robot.

IV. Simulation

In order to examine adequately the capability of the mobile robot, some tasks are performed in simulations by designing a robot simulator in Gazebo.

A. Experiment Setup

An Omni robot is considered as an agent. The simulation constructed in a static map shown as Fig. 5. The desired area is spawned randomly as the target area and a goal of agent. When reaching the destination or colliding to the wall, the reward was set +40 or -20 respectively.

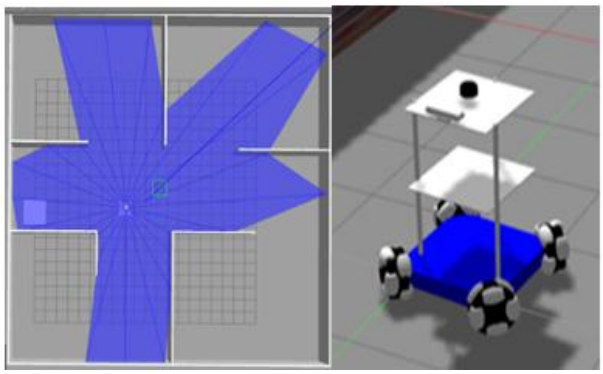

Figure 5. Robot model and environment in Gazebo.

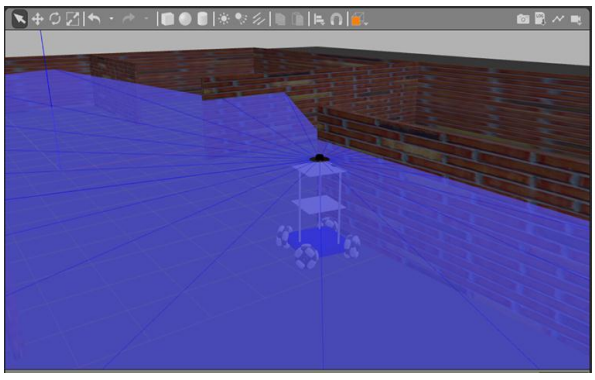

Figure 6. Robot model occur collision.

The robot model is designed using the URDF package, the Astra camera and the RPLidar are placed on the top of the robot while Omni wheels are designed with some rollers arranged around the perimeter of the wheel, which allows the Omni robot can move in all directions.

At the initial time, the robot model is placed on the generated map shown in Fig. 6 with some walls built in Gazebo. The blue area depicts the laser scan signal generated from RPLidar that used to identify the obstacle around the Omni robot.

B. Simulation Result

The Omni robot learns when a collision occurs, then it returns to the initiative position and adjust the net weight again based on the former learning result. From the result, when the robot is not high-understanding of the surrounding, the robot trajectories at early stages is not smooth. It takes a long time to acquaint the environment and reach toward the goal. After 300 epochs, the robot is capable of obstacle avoidance and has tendency toward the goal shown as Fig. 7. The Fig. 8 shows that the Omni robot perceives the shorter trajectory of toward goal with after approximately 700 epochs.

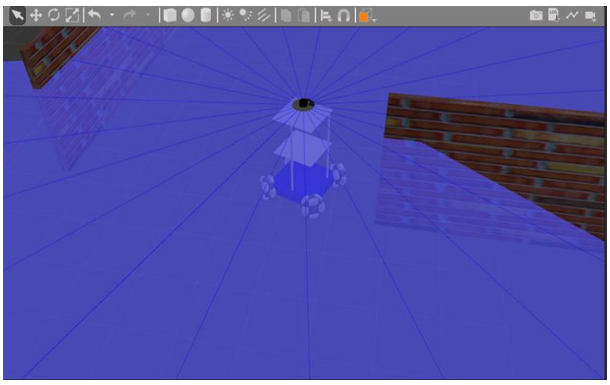

Figure 7. The ability of Obstacle avoidance.

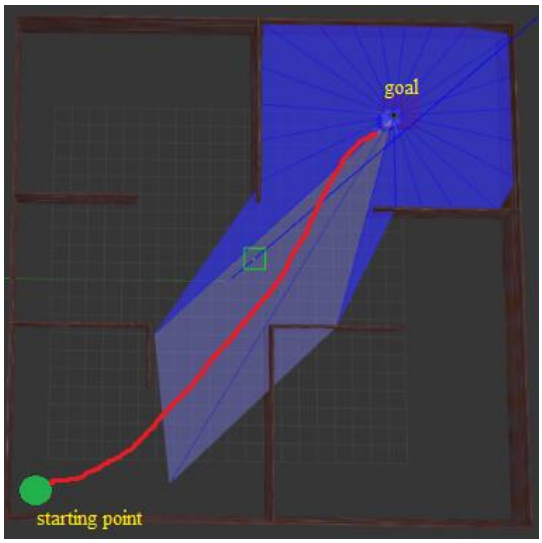

Figure 8. The ability of Obstacle avoidance.

RVIZ is a powerful tool in ROS to visualize and monitor the movement of the robot as well as the values of the sensors. The red lines show the range of the laser

scan which is published by Rplidar in Fig. 9. The robot model which is construccted in Gazebo is also monitored with its coordinates. Moreover the coordinate and the position of the robot are easy to observe.

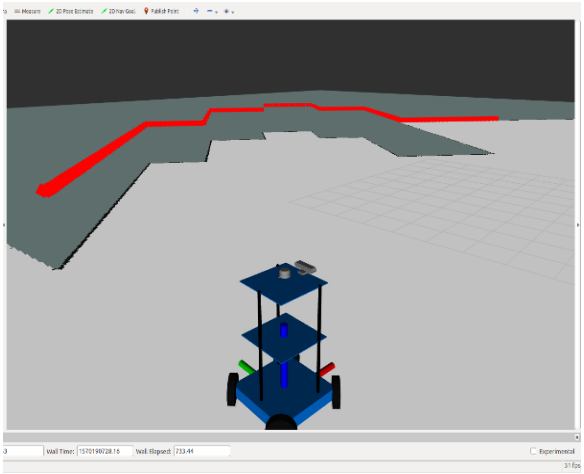

Figure 9. Robot movement in monitoring software RVIZ.

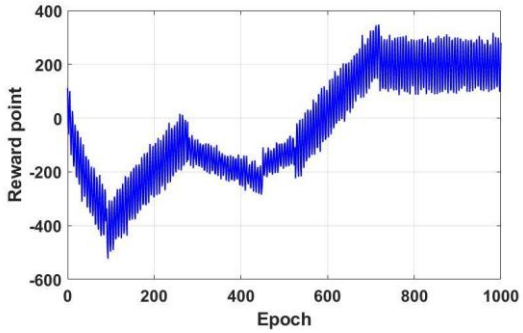

Figure 10. The total rewards of each epoch.

Fig. 10 shows the total reward obtained by the robot in each epoch. During the initial period, when the Omni robot inadequately gets information on the environment, there are two tendencies in the robot movement. The first trend of that is the collision with the surround obstacles due to the knowledge shortage of the robot about the landmarks. The remaining trend is the robot's motion described by randomly navigating in any directions and this motion is performed based on the defined robot' actions. Therefore, the robot is not likely to reach the goal position so that the total reward points fluctuate around some negative values. After a hundred epochs, the states of environment gradually are identified by the designed reinforcement network, leading to an upward trend of the reward point as well as the robot has the ability to move to the reference position. When the robot's knowledge reaches a certain threshold, the robot is capable to move smoothly to the desired goal, that leads to the increase to positive values of the reward point as verifying the effectiveness of the proposed method.

V. Conclusion

This paper has presented the construction of the navigation task for the four-wheeled Omnidirectional mobile robot and proposed the reinforcement learning method for implementing the robot's operating process in the virtual map built in Gazebo. The robot system and the reinforcement network are built in ROS. The task are based on the network with the laser scan signal generated from RPLidar is the input data. The simulation results show the significant performance of the proposed method in obstacle avoidance and finding the path toward the goal destination for the Omni robot. Moreover, the robot's operation could be monitored through the visualization tool and opens high likelihood DQN algorithm into differently specific environments. The practical model for the Omni robot will be constructed and the reinforcement learning network will be implemented in the robot's navigation task in the real environments for the future work.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Van Nguyen Thi Thanh, Tien Ngo Manh and Cuong Nguyen Manh designed the proposed algorithm; Dung Pham Tien and Duy Nguyen Duc conducted the experiment; Van Nguyen Thi Thanh, Tien Ngo Manh, Cuong Nguyen Manh and Manh Tran Van wrote the paper.

Acknowledgement

This research was funded by Project implemented by the Department of Image Technology and Automation, Institute of Physics, Vietnam Academy of Science and Technology

References

[1] A. H. Qureshi, Y. Nakamura, Y. Yoshikawa, and H. Ishiguro, "Robot gains social intelligence through multimodal deep reinforcement learning," *IEEE-RAS Int. Conf. Humanoid Robot.*, pp. 745–751, 2016.

[2] A. Folkers, M. Rick, and C. Buskens, "Controlling an autonomous vehicle with deep reinforcement learning," *IEEE Intell. Veh. Symp. Proc.*, vol. 2019-June, pp. 2025–2031, 2019.

[3] M. Kuderer, S. Gulati, and W. Burgard, "Learning driving styles for autonomous vehicles from demonstration," in *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2015-June, no. June, pp. 2641–2646, 2015.

[4] V. Mnih *et al.*, "Playing Atari with Deep Reinforcement Learning," p. https://arxiv.org/abs/1312.5602, 2013.

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

[6] A. Nair *et al.*, "Massively parallel methods for deep reinforcement learning," *arXiv preprint arXiv:1507.04296*, 2015.

[7] S. Amarjyoti, "Deep reinforcement learning for robotic manipulation-the state of the art," *Bull. Transilv. Univ. Braşov*, vol. 10, no. 2, 2017.

[8] A. V. Bernstein, E. Burnaev, and O. Kachan, "Reinforcement learning for computer vision and robot navigation," in *Proc. International Conference on Machine Learning and Data Mining in Pattern Recognition*, 2018, pp. 258-272: Springer.

[9] V. Matt and N. Aran, "Deep reinforcement learning approach to autonomous driving," ed: arXiv, 2017.

[10] X. Da and J. Grizzle, "Combining trajectory optimization, supervised machine learning, and model structure for mitigating the curse of dimensionality in the control of bipedal robots," *Int. J.*

Rob. Res., vol. 38, no. 9, pp. 1063–1097, 2019.
[11] I. Zamora, N. G. Lopez, V. M. Vilches, and A. H. Cordero, "Extending the openai gym for robotics: A toolkit for reinforcement learning using ros and gazebo," *arXiv preprint arXiv:1608.05742*, 2016.
[12] H. Kretzschmar, M. Spies, C. Sprunk, and W. Burgard, "Socially compliant mobile robot navigation via inverse reinforcement learning," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1289-1307, 2016.
[13] L. Tai and M. Liu, "A robot exploration strategy based on q-learning network," in *Proc. 2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, 2016, pp. 57-62.
[14] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Proc. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 31-36.
[15] L. Xie, S. Wang, A. Markham, and N. Trigoni, "Towards monocular vision based obstacle avoidance through deep reinforcement learning," *arXiv preprint arXiv:1706.09829*, 2017.

Van Nguyen Thi Thanh Graduated Engineering Degree major in Electronic Engineering at Hanoi University of Industry (HaUI) from 2006-2010. Defensed Master Degree in Control Engineering and Automation at Military Technical Academy in 2014. Now, works at Hanoi Vocational College of High Technology. The main researches: Process control, PLC controller and industrial communication network, adaptive control, fuzzy logic, and neural network control.

Tien Ngo Manh Graduated Engineering Degree major in Automatic Control at Hanoi University of Science and Technology (HUST) from 1996-2001. Defensed Dr. Degree in Electrical Engineering at HUST in 2014. Now, works at Institute of Physics, Vietnam Academy of Science and Technology. The main researches: Process control, adaptive control, fuzzy logic and neural network control, automatic robot control, electro-optical system, image processing.

Duyen Ha Thi Kim Graduated Engineering Degree major in Automatic Control at Hanoi University of Science and Technology (HUST) from 1996-2001. Defensed the Master Degree at Le Quy Don Technical University in 2007. Now, works at School of Electronic - Hanoi University of Industry. The main researches: Process control, PLC controller and industrial communication network, adaptive control, fuzzy logic, and neural network control.

Cuong Nguyen Manh senior student major in Electrical – Automatic Control at Hanoi University of Science and Technology (HUST). Now, working at Institute of Physics, Vietnam Academy of Science and Technology. The main researches: Adaptive control, fuzzy logic and neural network control, Deep learning and Robot Operating System programing for robotics.

Manh Tran Van senior student major in Electrical – Automatic Control at Hanoi University of Science and Technology (HUST). Now, working at Institute of Physics, Vietnam Academy of Science and Technology. The main researches: Adaptive control, fuzzy logic and neural network control, Deep learning and Robot Operating System programing for robotics.

Dung Pham Tien senior student major in Electrical – Automatic Control at Hanoi University of Science and Technology (HUST). Now, working at Institute of Physics, Vietnam Academy of Science and Technology. The main researches: Adaptive control, fuzzy logic and neural network control, Deep learning and Robot Operating System programing for robotics.

Duy Nguyen Duc student major in Computer Engineering at Hanoi University of Industry. Now, working at Institute of Physics, Vietnam Academy of Science and Technology. The main researches: Deep learning and Robot Operating System programing for robotics.