Asynchronous Deep Reinforcement Learning for the Mobile Robot Navigation with Supervised Auxiliary Tasks

T. Tongloy[1], S. Chuwongin[2], K. Jaksukam[3]
Center of Industrial Robots and Automation (CiRA)
College of Advanced Manufacturing Innovation (AMI)
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
e-mail:[1]teerawat.tongloy@gmail.com, [2]santhad.ch@kmitl.ac.th , [3]jkomgrit@gmail.com

C Chousangsuntorn[4], S. Boonsang*[5]
Department of Electrical Engineering,
Faculty of Engineering,
King Mongkut's Institute of Technology Ladkrabang
Bangkok, Thailand
e-mail: [4] chousak.jantaco@seagate.com, [5] siridech.bo@kmitl.ac.th

***Abstract*—In this paper, we present the method based on asynchronous deep reinforcement learning adapted for the mobile robot navigation with supervised auxiliary tasks. We apply the hybrid Asynchronous Advantage Actor-Critic (A3C) algorithm CPU/GPU based on TensorFlow. The mobile robot is simulated as the navigation tasks on the OpenAI-Gym-Gazebo-based environment with the collaboration with ROS Multimaster. The supervised auxiliary tasks include the depth predictions and the robot position estimation. The simulated mobile robot shows the capability to learn to navigate only the input from raw RGB-image and also perform recognition of the place on the map. We also show that the combination of all possible auxiliary tasks leads to the different learning rate.**

Keywords-component; asynchronous reinforcement learning; GA3C; ROS; supervised auxiliary tasks; mobile robot navigation

I. INTRODUCTION

The skill to navigate and recognize the place within the complex environment is essential to robot's intelligent behavior approaching human-level capability. Typically, the conventional robot navigation methods, such as Simultaneous Localization and Mapping (SLAM), tackle this challenge through the reconstruction of an obstacle map of the navigation environment using data from dense laser range finders. Associated features of obstacles and navigation paths are extracted to pinpoint the robot position and assist the robot to navigate to the predefined destination. However, there are two main difficulties involving the navigation using SLAM. These include the laborious, expensive and time-consuming map construction and updating processes and also the demand for the precise laser range finder to achieve the high-quality obstacle map. Therefore, it is still a substantial requirement for the method to describe the appropriate navigation behaviors for mobile robots without an obstacle map.

In the last few years, Deep Reinforcement Learning (DRL) has successfully demonstrated the capability of Convolutional Neural Networks (CNNs) to estimate value and policy functions as the part of Reinforcement Learning process (RL). With this approach, a series of progression has been revealed. These include the Deep Q-Learning Network (DQN) algorithm [1] for learning to play the classical video games such as Atari with simply input from raw pixels [2-3] and the algorithm so-called AlphaGo in which it demonstrates the extraordinary performance over the best human player on the board game Go [4] In addition, a variety of compelling learning methodology and CNN architectures [5-6] are also proposed not only to achieve the best learning rate but also to increase parallelism while decreasing the computational cost and memory burdensome [7-8]. Specifically, Mnih et al. demonstrated a parallel method so-called Asynchronous Advantage Actor- Critic (A3C) in which it can achieve exceptional results on many gaming tasks with relatively lightweight [8].

There are several efforts to apply the DRL techniques for mobile robot navigation. Zhu et al. [9] formulated the navigation problem using A3C by introducing the AI-THOR platform integrated a well-known physics engine (Unity 3D) with a deep learning framework of Tensorflow. Such an integration enables inexpensive and efficient collection of action and interaction data. They also validated the model by using the real-world environment. However, the predefined discrete action space to simplify the task might have disadvantages of the robot agility and cumbersome for the simulated environment construction. Mirowski et al. [10] proposed the model so-called "NavA3C" model with supervised auxiliary tasks including the depth prediction and loop closure prediction tasks. Their result shows that the NavA3C with supervised auxiliary tasks significantly enhances the overall learning performance for navigation of agent. However, the application of such platform is limited

978-1-5386-1306-1/17/$31.00 ©2017 IEEE

to the gaming environment which is difficult to transfer to the real world environment. Recently, the algorithm of the asynchronous advantage actor-critic on a GPU or "GA3C" [11] which is the modification of the A3C has shown the superior performance over the original the A3C. The main mechanism of GA3C comprises of many predictors to query the CNN model for policies and trainers to collect actor's experiences (occurred when actors interact with the environment) and send associated information to CNN model to compute the gradients for the CNN model optimization routine. The GA3C utilizes the superior mathematical performance of a GPU and also uses only one copy of the shared CNN model for all trainers and agents whereas the A3C must replicate a CNN model for each agent.

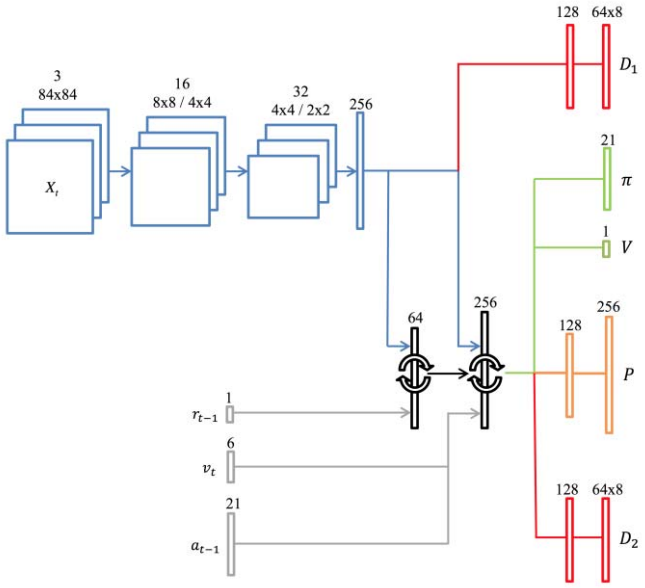


Figure 1. The NavGA3C model with supervised auxiliary tasks D_1, D_2 and P that modified from NavA3C [10].

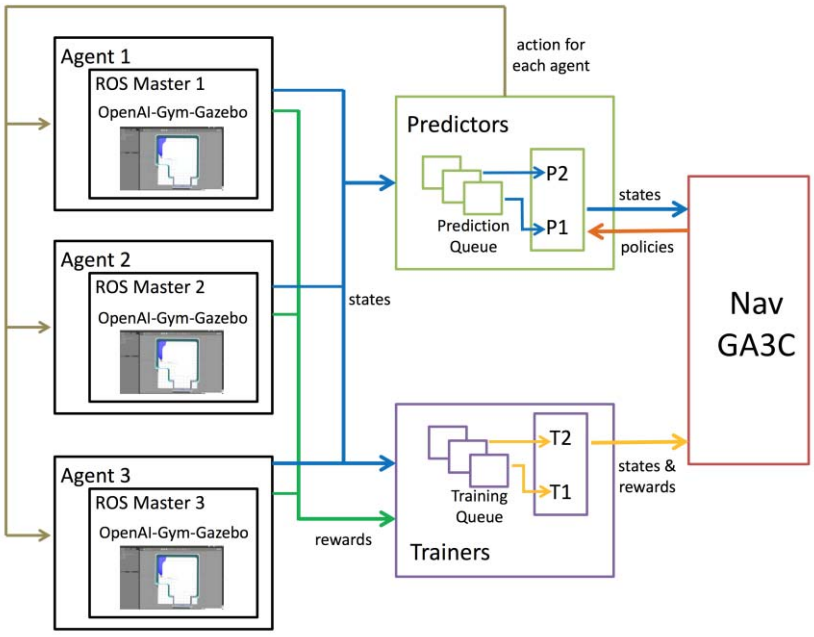


Figure 2. Training setup of Nav GA3C model and OpenAI-Gym-Gazebo with ROS Multimaster (3 agents, 2 trainers and 2 predictors).

TABLE I. GRADIENT SCALE VALUES OF AUXILIARY TASKS

Model	B_{d1}	B_{d2}	B_p
NavGA3C	-	-	-
NavGA3C+D_1	1	-	-
NavGA3C+D_2	-	1	
NavGA3C+P	-		1
NavGA3C+D_1D_2	10	5	-
NavGA3C+D_1P	10	5	-
NavGA3C+D_2P	-	1	1
NavGA3C+D_1D_2P	10	3.33	3.33

TABLE II. GRADIENT SCALE VALUES OF AUXILIARY TASKS

Model	B_{d1}	B_{d2}	B_p
NavGA3C	-	-	-
NavGA3C+D_1	1	-	-
NavGA3C+D_2	-	1	
NavGA3C+P	-		1
NavGA3C+D_1D_2	10	5	-
NavGA3C+D_1P	10	5	-
NavGA3C+D_2P	-	1	1
NavGA3C+D_1D_2P	10	3.33	3.33

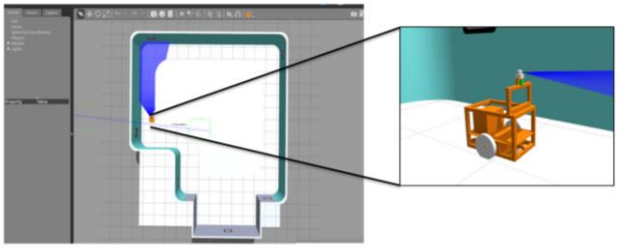
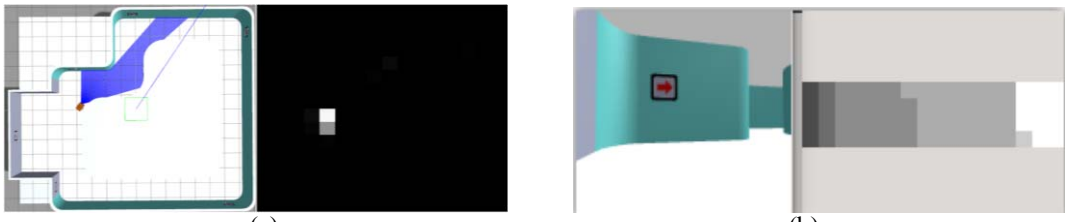
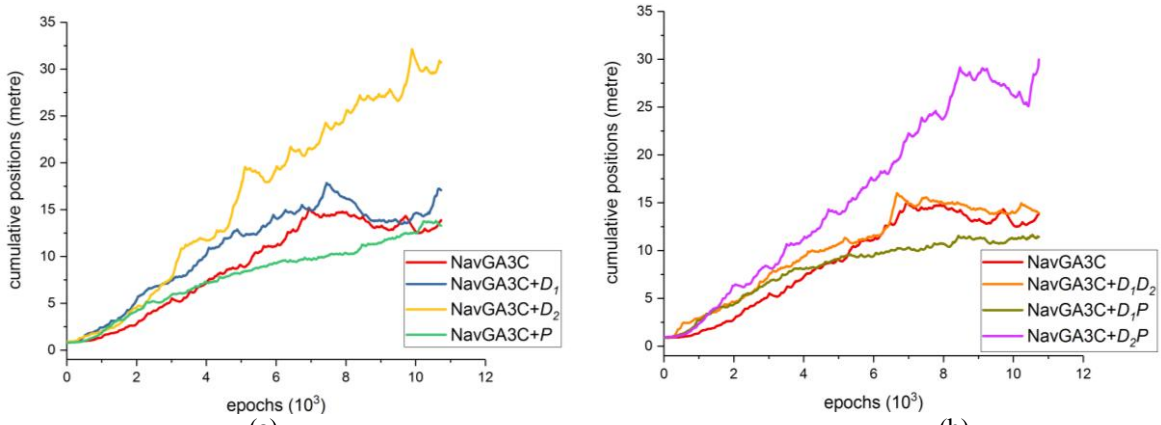


Figure 3. Map and the mobile robot at the start position in Gazebo simulator.



(a) (b)

Figure 4. (a) the position prediction result (b) the depth prediction result.



(a) (b)

Figure 5. Reward achieved from the different model configurations (a) comparing the case of the model contains only one auxiliary task and the model with no auxiliary task (b) the learning curves comparing the case of the model contains two auxiliary tasks and the model with no auxiliary task.

In this paper, we propose the DRL technique adapted from the GA3C algorithm for the mobile robot navigation with the extra supervised auxiliary tasks (the depth predictions and the position estimation). In addition, we also introduce the simulation platform which is the integration of the OpenAI-Gym and Robot Operating System (ROS) with Gazebo simulator. ROS is a popular framework for robotics development and research. The main advantage of using ROS as the simulation platform is the facility for transferring the simulation model to the real world environment. In addition, we also compare the learning rate effect of adding individual auxiliary tasks into the NavGA3C model.

II. Asynchronous Deep Reinforcement Learning Model

In this section, the network architectures and training setup for Asynchronous DRL for the mobile robot with supervised auxiliary tasks are described extensively as follows:

A. The Modified NavGA3C Model

The NavGA3C model is shown in Fig. 1. The NavGA3C model consists of the main Convolutional Neural Network (CNN) encoder and two-layers stacked LSTM. An RGB image X_t is fed to the encoder as input. In addition, a previous reward r_{t-1} is also used as input to the first LSTM layer and a robot velocity V_t with previous action a_{t-1} are also configured as the input to the second LSTM. Main outputs of this network are the approximation of a policy function π for actors and the corresponding value function V. In order to simplify the action task, the action space is discretized into 21 actions over the forward angle of the mobile robot (-1.05 rad to 1.05 rad).

For supervised auxiliary tasks, the first task is depth prediction D_1 directly connected from the CNN encoder and D_2 linked to the output from the two-layers stacked LSTM. These tasks are constructed as the classification function. The objective is to provide the robot with the prediction of forwarding direction of low-resolution depth image (4x16 pixels). The prediction of the depth is in the form of discrete values up to 8 levels. The loss function calculation of both tasks is independent of the main policy and value functions. However, the gradient calculation for CNN optimization routine has also accounted to the auxiliary these tasks via the factors B_{d1} and B_{d2}. The ground truth values for these tasks are acquired by the measured RGBD values of the robot.

The last auxiliary task is robot position prediction relative to navigation map. This is also configured to the classification function to predict position P of the robot in the map. The map is discretized into 256 positions by 16x16 grid. The loss function calculation is the same concept as the previous depth prediction as described above. The ground truth values are assigned by using the position reported from the environment. The factor B_p is used for the contribution of this loss values to the main CNN optimization routine.

B. The Simulation Environment: OpenAI-Gym-Gazebo with ROS Multimaster

We adapt the OpenAI-Gym-Gazebo which is a toolkit for developing and comparing reinforcement learning algorithms using OpenAI-Gym, ROS and Gazebo [12]. This toolkit is originally developed in the OpenAI-Gym to interact with a single ROS master, which establishes the connection between the Gym itself and Gazebo simulator. The Gazebo provides a robust physics engine with high-quality graphics and graphical interfaces. However, the GA3C algorithm typically utilizes more than one agents asynchronously in the learning process, therefore the number of environments to interact with agents must be equal to a number of simulated agents. In order to accommodate this requirement, we propose OpenAI-Gym-Gazebo with ROS Multimaster framework. The diagram in Fig. 2 illustrates the link between OpenAI-Gym, ROS Multimaster and Gazebo. With the ROS Multimaster concept, it is possible for each agent to interact with their own Gazebo engine which is independent run on individual ROS master. Therefore we can effectively run more than one Gym-Gazebo process on the same computer. In addition, ROS-Gazebo simulation also provides data necessary for ground truth value labelling. These include the depth data from laser range finders and also the position of the robot in the map.

C. Training Setup

For the setup of the training process, we utilize the GA3C architecture with 3 agents, 2 trainers and 2 predictors as shown in Fig. 2. All agents submit their current states to the predictors. The predictors process the agent's state and return the predicted actions and policy's values. After the agents perform their actions, they forward new states and rewards to the trainers. The trainers collect every states and reward in training queue as a batch. When batch size equals 50 batches, the trainers set forth all data in the batches to the modified NavGA3C model for computing the gradients and updating the CNN model. We use RMSProp algorithm with no momentum and learning rate 0.0005 for the optimization process.

The command velocity is discretized into 21 actions within the angular velocity in Z axis direction (left and right) between -1.05 rad/s to 1.05 rad/s. The translation velocity for all action in X axis (forward) is 0.2 m/s. Observation states of each step include an RGB image, previous reward, velocity and previously selected action. In addition, the environment provides the depth image and the robot position on the map for ground truth labelling for the supervised auxiliary tasks. The goal for the DRL is to provide the robot navigation model that can move as far as possible and perform obstacle avoidance. The reward is calculated from Lidar sensor to estimate the distance from the wall. Lidar sensor range is equal FOV of the camera. Accumulative distance is defined by a distance that mobile robot moves from the start position in each episode. Each model chooses gradient scales value of auxiliary tasks by considering the ratio between the number of tasks from the convolutional encoder and the tasks from stacked LSTM as shown in Table I.

III. Results and Discussions

Fig. 3 illustrates the simulation map and the mobile robot at the starting position in Gazebo simulator. The robot is set out from this position and wanders along the corridor. The simulation for each epoch stops and restarts again, once the robot hits the side wall. Fig. 4a shows the position prediction result (the white square dot in the black background) comparing with the top view showing the actual position of the robot within the map. Good agreement between the actual position and the predicted one is found. Fig. 4b demonstrates the depth prediction result. The low-resolution depth image correctly shows the depth corresponded to the actual depth shown on the left-hand side.

Fig. 5 shows the reward accumulated for each model configurations. In the case of the modified NavGA3C with 1 auxiliary task (Fig. 5a), NavGA3C+D_2 has more effective than others. It is clear that the knowledge about depth information in stacked LSTM enhances the capability of the mobile robot to efficiently move in an environment with obstacle avoidance. The NavGA3C+D_1 is the second most effective in which it learns faster than the NavGA3C at the beginning, however, after 6500 epochs, the learning curves are saturated and closed to each other. The NavGA3C+P is the model for learning the robot position in the map without the knowledge about depth information, it shows the least effective among all algorithm. This result suggests that, in order to train the robot to remember the place on the map, the depth information is vital.

In the case of the NavGA3C with 2 auxiliary tasks (Fig. 5b), NavGA3C+D_2P demonstrates the most effective algorithm. Comparing with the NavGA3C+D_1D_2 which is the second most effective, it is clear that the position prediction task P provides the learning rate enhancement to the NavGA3C+D_2. For the NavGA3C+D_1D_2 which learn faster than NavGA3C in the beginning, after 6500 epoch they are saturated close to each other. NavGA3C+D_1P has lower effective than NavGA3C.

IV. Conclusions

We introduce asynchronous deep reinforcement learning for the mobile robot navigation with supervised auxiliary tasks (depth predictions and position estimation). To learn and evaluate models, we modified the NavA3C by using GA3C as baseline architecture so-called the NavGA3C model. The OpenAI-Gym-Gazebo with ROS Multimaster is used to create the simulated environment. In addition, we also compare each model by several different combinations of auxiliary tasks to the NavGA3C model. The experimental results show that NavGA3C model with depth prediction from stacked LSTM (D_2) is the best auxiliary task in which it increases learning performance for the mobile robot to move in an environment with avoiding an obstacle. Depth prediction from the convolutional encoder (D_1) increases relatively low learning performance. In the case of adding only position auxiliary task (P) to stacked LSTM, the result indicates that the learning performance becomes worst. When combining D_2P to NavGA3C model, the learning performance shows a similar to the case that the model adding only D_2 to NavGA3C. In conclusion, NavGA3C+D_2P is the best performing model in our experiments because it provides the depth information and estimates robot position without decreased learning performance.

Acknowledgment

The work has been supported by research grants from the Seagate Technology (Thailand), College of Advanced Manufacturing Innovation (AMI), King Mongkut's Institute of Technology Ladkrabang and Thailand Research Fund (TRF).

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G.Bellemare, A.Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, St. Petersen,C. Beattie, A. Sadik, I. A., H. King, D. Kumaran, D.Wierstra, S. Legg, and D. Hassabis. "Human-level control through deep reinforcement learning". Nature, 518(7540):529–533, 02 2015. URL http://dx.doi.org/10.1038/ nature14236.

[2] M. G. Bellemare, S. Srinivasan, G. Ostrovski, T. Schaul, D. Saxton, and R. Munos. "Unifying Count-Based Exploration and Intrinsic Motivation". ArXiv e-prints, June 2016.

[3] G. Lample and D. Singh Chaplot. "Playing FPS Games with Deep Reinforcement Learning". ArXiv e-prints, September 2016.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Driessche, J Schrittwieser, I Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M.e Leach, K.y Kavukcuoglu, T.e Graepel, and D. Hassabis. "Mastering the game of go with deep neural networks and tree search". Nature, 529:484–503, 2016. URL http: //www.nature.com/nature/journal/v529/n7587/full/nature16961.html.

[5] H. Hasselt, A. Guez, and D. Silver. "Deep reinforcement learning with double qlearning". CoRR, abs/1509.06461, 2015. URL http://arxiv.org/abs/1509.06461.

[6] Z. Wang, N. Freitas, and M. Lanctot. "Dueling network architectures for deep reinforcement learning". CoRR, abs/1511.06581, 2015. URL http://arxiv.org/abs/1511. 06581.

[7] A. Nair, P.Srinivasan, S. Blackwell, C. Alcicek, R. Fearon, A.o De Maria, V. Panneershelvam, M. Suleyman, C. Beattie, S. Petersen, S. Legg, V. Mnih, K. Kavukcuoglu, and D. Silver. "Massively parallel methods fordeep reinforcement learning". CoRR, abs/1507.04296, 2015. URL http://arxiv.org/abs/ 1507.04296.

[8] V. Mnih, A. Puigdomenech Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. "Asynchronous Methods for Deep Reinforcement Learning". ArXiv preprint arXiv:1602.01783, 2016.

[9] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," arXiv preprint arXiv:1609.05143, 2016.

[10] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu et al., "Learning to navigate in complex environments," arXiv preprint arXiv:1611.03673, 2016.

[11] M. Babaeizadeh, I. Frosio, S. Tyree, J. Clemons, J. Kautz, "reinforcement learning through asynchronous advantage actor-critic on a GPU" ICLR 2017

[12] I. Zamora, N. Gonzalez L.Victor,M. Vilches, A. H. Cordero, "Extending the OpenAI Gym for robotics: a toolkit for reinforcement learning using ROS and Gazebo" arXiv:1608.0574