Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A.A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

SIMULATION ANALYSIS OF A DEEP REINFORCEMENT LEARNING APPROACH FOR TASK SELECTION BY AUTONOMOUS MATERIAL HANDLING VEHICLES

Maojia Patrick Li
Prashant Sankaran
Michael E. Kuhl

Industrial and Systems Engineering Department
Rochester Institute of Technology
Rochester, NY 14623, USA

Amlan Ganguly
Andres Kwasinski
Raymond Ptucha

Computer Engineering Department
Rochester Institute of Technology
Rochester, NY 14623, USA

ABSTRACT

The use of autonomous vehicles is a growing trend in the material handling and warehousing. Some challenges that face material handling include the navigation within a warehouse, precision localization and movement, and task selection decisions. In this paper, we address the issue of task selection. In particular, we develop a deep reinforcement learning methodology to enable a vehicle to select from among multiple tasks and move to the closest task in the context of material handling in a warehouse. To evaluate the deep reinforcement learning methodology, we conduct a simulation-based experiment to generate scenarios to first train and then test the capabilities of the method. The results of the experiment show that the method performs well under the given conditions.

1 INTRODUCTION

The use of autonomous vehicles is a growing trend in the material handling and warehousing. Companies have used Automated Guided Vehicles (AGVs) for decades where vehicles travel along a guided path making scheduled stops to pick-up or drop-off items. However, as technology has advanced, a new generation of material handling equipment is being developed. In particular, the automated technologies that enable self-driving cars, collision avoidance systems for the automotive industry are beginning to be applied to material handling. Some challenges that face material handling that are different than the automotive industry include the navigation of vehicles indoors (without GPS), precision localization and movements, and task selection decisions, among others.

To address the issues of control and decision-making of autonomous vehicles in the context of material handling, we are developing what we call an intelligent Material Handling (iMH) system which we feel will represent the next major step in the advancement of warehouse productivity and safety. iMH integrates mapping and localization, sensors, computer vision, communication, and decision-making for autonomous vehicles in a warehouse setting. As part of this effort, we are developing machine/deep-learning approaches to enable adaptability of a fleet of autonomous vehicles to a facility and refined situational decision-making. In particular, we are interested in the problem of task selection. That is, if a fleet of vehicles has a list of tasks to complete, how can vehicles autonomously determine task selection to complete the list of tasks as efficiently as possible? Given this goal, the focus of this paper is on task selection for a single vehicle that is given multiple alternatives to select from among. We develop and conduct a simulation study on a deep reinforcement learning methodology in the context of material handling to enable a vehicle to select and move to the closest task given the layout and obstacles (racks, pallets, etc.) of a warehouse.

The remainder of the paper is organized as follows. In Section 2, we present a review of related work. In Section 3, we develop the deep-learning methodology. We discuss the simulation-based experiment in Section 4 and the results in Section 5. Finally, conclusions and future work are presenting in Section 6.

2 RELATED WORK

Much of the available research on autonomous vehicle systems focuses on vehicle navigation and path planning. The early industrial autonomous vehicles, which are usually referred as automated guided vehicles (AGV) are navigated by physical guide path such as, buried wire or magnetic tape. Such a system requires floor alternations and reduces the flexibility of vehicles. The second generation of autonomous vehicles use wireless guidance systems, such as laser and inertia systems, to overcome these issues. Thanks to the recently advanced deep learning and computer vision techniques, we now have the next generation of autonomous vehicles, which rely on vision-based guidance systems. These vehicles can localize themselves and track pre-determined paths with a minimal level of supervision. Researchers have been trying to implement vision guidance systems on autonomous vehicles since the early 2000s. Fang and Xie (2004) combines three navigation modules to help autonomous vehicles follow a designated path. The main, conflict-free, and special navigation modules are used for lane recognition, obstacle detection, and special landmark identification. Ramer et al. (2015) implemented a data fusion technique to integrate information collected by cameras, odometer, LIDAR, and collision avoidance sensors for the purpose of navigation.

As autonomous vehicles are no longer constrained by a guide path or artificial landmarks, it becomes more challenging for the vehicles to find an efficient and safe path to the destination. The most common approach is to convert the floor plan into a grid map so that the vehicle will search for the shortest, obstacle-free path. Martínez-Barberá and Herrero-Pérez (2010) compare A* (Hart et al. 1968) and D* (Stentz 1994) algorithms for autonomous vehicle path planning within a partially structured warehouse. Jeon et al. (2011) propose a routing method for autonomous vehicles in container terminals using a Q-learning algorithm. Herrero-Perez and Martinez-Barbera (2010) introduce a hierarchical approach to reduce the computational cost for path planning. The two-level approach uses a topology map to store high-level description of the environment and grid maps for local information. Another hierarchical approach proposed by Contreras-Cruz et al. (2015) combines an artificial bee colony algorithm and an evolutionary programming algorithm.

Despite the considerable amount of recent research on autonomous vehicle navigation and path planning, few researchers have focused on the task assignment problem. Egbelu and Tanchoco (1984) classifies autonomous dispatching into vehicle-initiated and workstation-initiated problems. In a workstation-initiated problem, there are multiple available autonomous vehicles and one transportation request. In contrast, a vehicle-initiated problem refers to the circumstances where multiple requests are waiting but only one autonomous vehicle is available. Egbelu and Tanchoco (1984) show that the performance of an autonomous vehicle system is mainly governed by the vehicle-initiated problem. Summaries of researches about vehicle-imitated problems can be found in several reviews (Le-Anh and De Koster 2006; Fazlollahtabar and Saidi-Mehrabad 2015). Two of the most popular dispatching methods are policy-based dispatching rule and optimization techniques. However, these approaches, for instance the shortest travel distance (STD) rule, assumes that the vehicle can easily determine the distance from its current location to each request's pickup location. Such an assumption is not held for vision-guided vehicles as it has a large number of options to move from one location to another. One possible approach is to utilize A* or reinforcement learning based technique to compute the distance from the vehicle to each potential destination and then compare their distances. However, such an approach will lead to high computational cost and the distance information of the unselected requests are usually discarded.

3 METHODOLOGY

We propose a deep reinforcement learning based decision-making framework where the system makes decisions based on the current situation or current state in a dynamic environment. The decision-making framework is represented in Figure 1. The autonomous agent will interact with the environment through control hardware and collects sensory inputs (current situation or state representations), which is preprocessed into state data, containing the current layout represented as a multi-dimensional array including the current location of the agent, tasks and obstacles in the layout. This state data is fed into the decision support system for task selection.

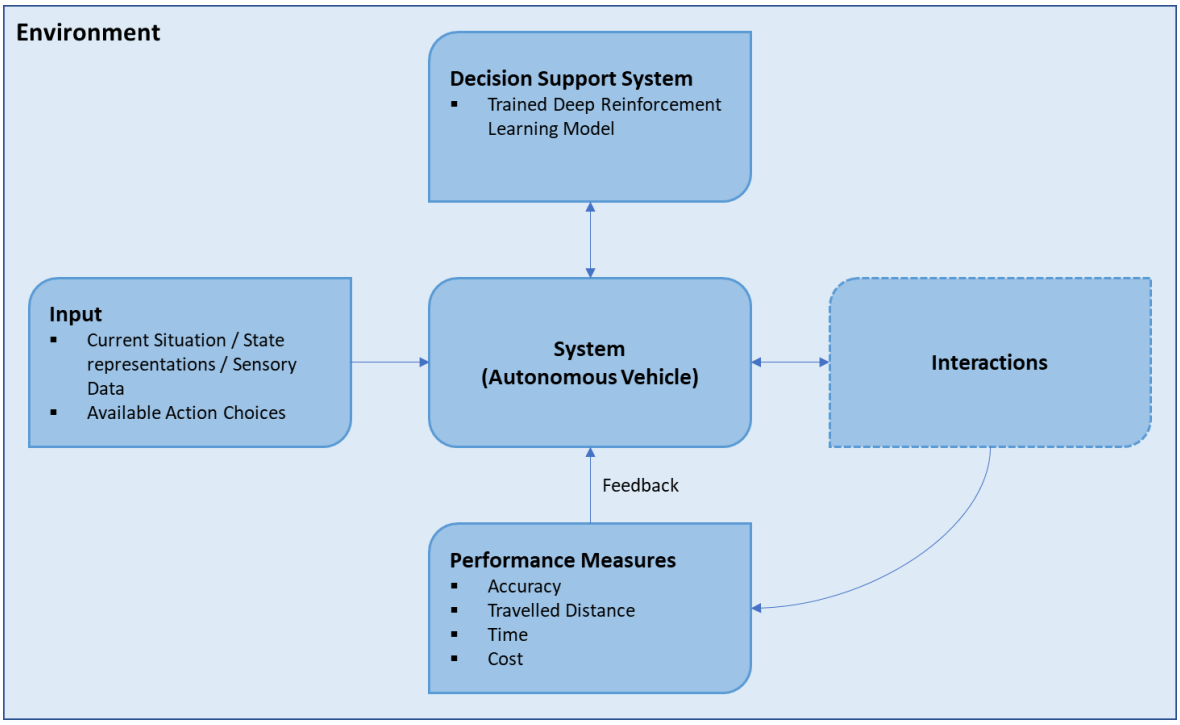


Figure 1: Deep reinforcement learning based decision-making framework.

The decision support system, illustrated in Figure 2, takes the state input and translates it into the virtual environment and performs a search operation to find a task by virtually moving the vehicle using available control. With regards to deep reinforcement learning the reward system design within the virtual environment is critical in enabling the agent to learn appropriate policies and improve the search process, thereby leading to selection of the best task. The virtual environment is exited once a task is selected and the result thus obtained is returned as an output of the decision support system. Further, the system will also be able to measure its own performance based on future state input and thus improve.

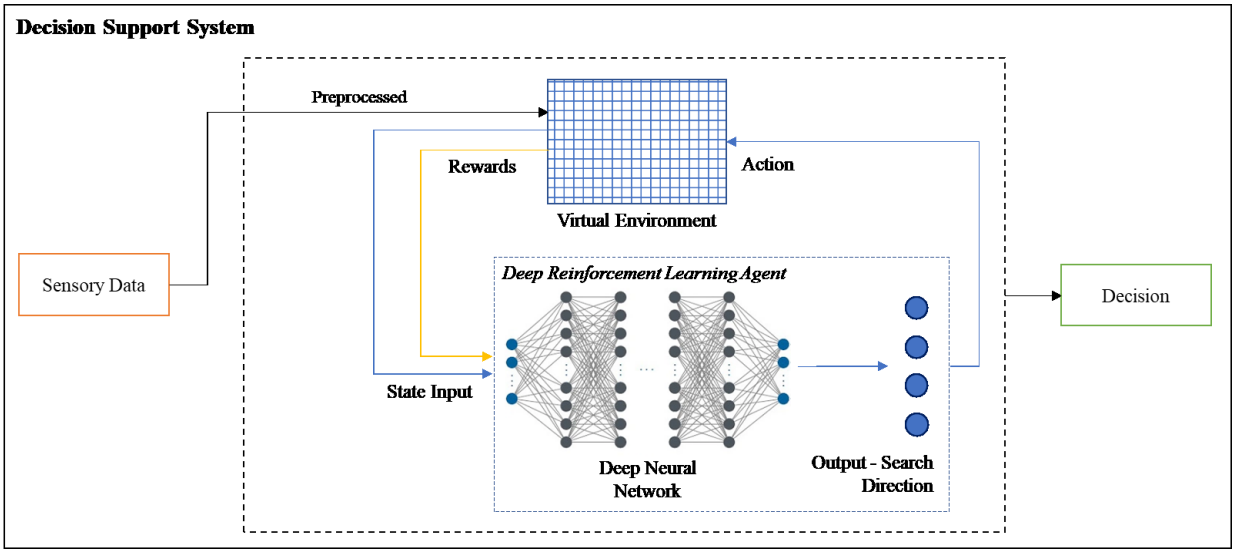


Figure 2: Illustration of the decision support system.

In the context of the warehouse environment, the autonomous vehicle will decide about the next task selection based on inputs about the current system state in terms of the list of available tasks, the locations of the vehicle and available tasks, etc. The vehicle will utilize the trained deep reinforcement learning model in combination with the virtual environment to render a task decision. The vehicle then executes the decided task. Upon completing the task, the performance of the decision can be analyzed and feedback provided to reinforce the learning process.

For a successful reinforcement learning implementation, the problem needs to be formulated as an Markov Decision Process (MDP), where in an MDP an agent in an environment at a given state will receive an observation (new state data) and needs to choose an action from a finite set of actions (search directions). The action moves the agent into a new state so that the agent will capture a new observation and receive a reward. The goal is to maximize the total reward over time. Given its characteristics, we hypothesize that the autonomous vehicle task selection problem can be formulated as an MDP.

Once we have a problem formulated as an MDP, we can use a suitable deep reinforcement learning architecture that uses a combination of convolutional neural network, and Q-learning to train the model using simulation. A deep convolutional neural network architecture is used to automate the learning process of filters that extracts the layout features such as corners, edges, task location, obstacles etc., given input layout data through a backward propagation of the convolutional layers, while a Q-learning is used to determine the appropriate search policy. An example of a deep convolutional neural network architecture containing both the convolutional, and fully connected layers for search direction prediction can be found in Figure 4. During the training process the exploration rate governs the extent of learning. The exploration rate is computed as,

$$\epsilon \equiv \frac{\epsilon_{max} - \epsilon_{min}}{n_{exp}} * n_{cur}$$

where the constants ϵ_{max}, ϵ_{min}, and n_{exp}, represent the maximum exploration rate, minimum exploration rate, and number of search iterations to explore, and the variable n_{cur} is the current search iteration. For example, if the exploration rate is 0.5, there is 50% chance to select a random action or search direction and rest 50% based on experience or action with a maximum Q value.

To illustrate the effectiveness of this approach to address the task selection problem in material handling, we conduct the following simulation-based experiment.

4 SIMULATION-BASED EXPERIMENT IN MATERIAL HANDLING CONTEXT

The framework developed addresses the need effective task selections decisions in a warehouse environment, where tasks arrive and depart at random time intervals and a large fleet of autonomous vehicles (agent) are used to execute the pickup and drop off tasks. To demonstrate the potential of our framework, we chose to conduct simulation experiments on a single agent to investigate the ability of the agent to learn and perform task selection based on a measure of shortest travel distance.

The simulation environment used consists of a generalized layout of 26 x 26 grid with each grid size equal to the size of the autonomous vehicle. Figure 3 illustrates the warehouse layout under consideration in the experiment. During each simulated replication, the agent is randomly placed in one of the pathway locations throughout the facility. The tasks are randomly placed in any of the 240 locations adjacent to a potential storage location. During a replication, if the current exploration rate ϵ is greater than 0, the replication is defined as an exploration replication. Otherwise, the replication is an exploitation replication, since all the actions are chose based on the highest state-action values.

The deep reinforcement learning model used in our simulation is Deep Q-Network (DQN), which was developed and advanced by Google DeepMind (Mnih et al. 2013; Mnih et al. 2015). The DQN algorithm is designed for Markov Decision Processes (MDP) and combines deep Convolutional Neural Network (CNN) and Q-learning.

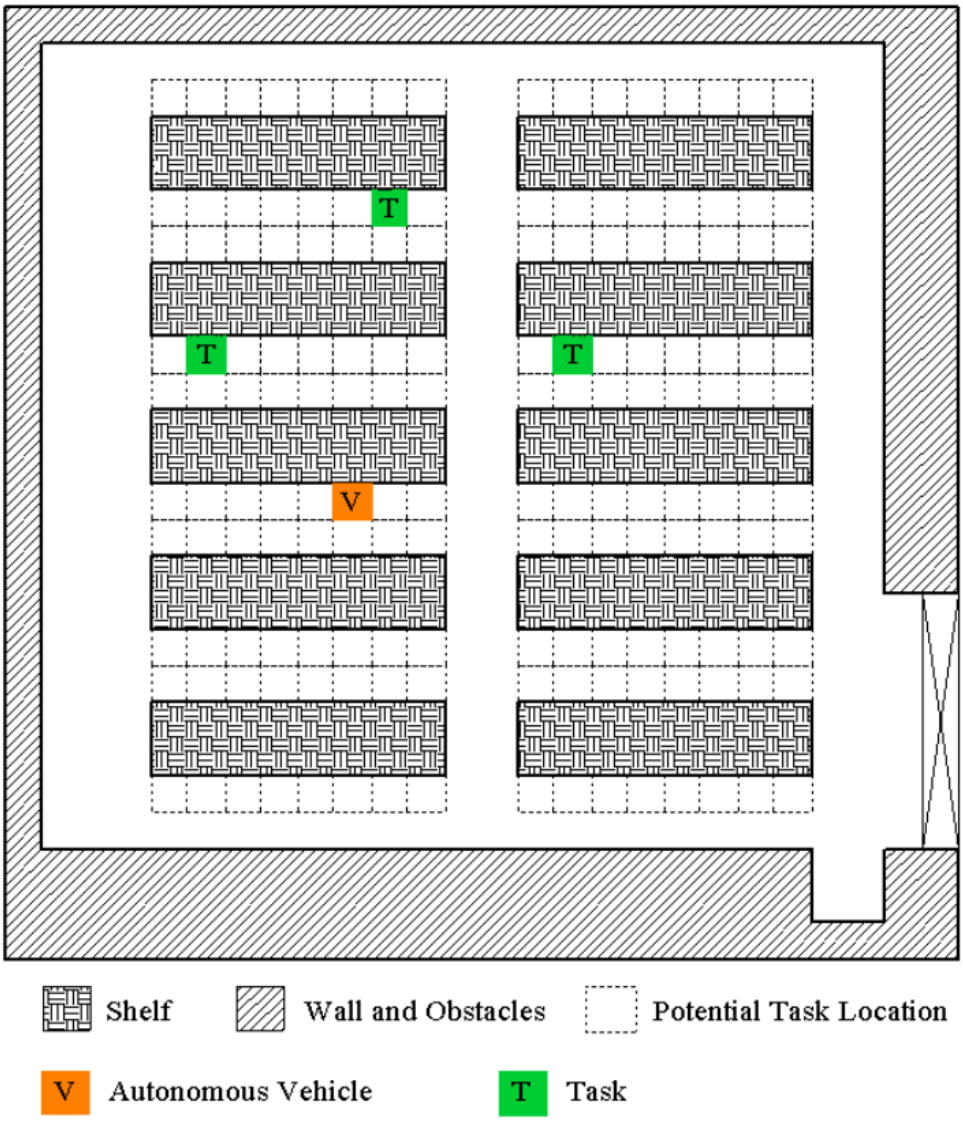


Figure 3: Layout depicting task locations (T), the autonomous vehicle (V) and walls and shelves marked as hatches and patterns, respectively.

The agent is trained using this simulated environment with a modified version of DQN that adapts to our specific problem. DQN uses a convolutional neural network as a function approximator to get a good estimate of state specific action choices that lead to maximum reward on an accrued basis. This method is computationally efficient in searching under unseen states as against traditional A* type methods. As a general guideline, learning algorithm become robust if the training is free from over or under estimation error. To achieve that we address the underestimation problem using large replications (about 1,000,000 action steps). Overestimation, specifically of the action values (Van Hasselt 2010) or search directions, is addressed by using DQN with double Q learning, which separates the policy estimates and its action values using two different value functions (Van Hasselt et al. 2016).

The network used for our application is given in Figure 4, which explains the deep convolutional neural network architecture. The input of dimension 26 x 26 (floor layout) and 4 consecutive state representations each containing the current location of the vehicle/agent, tasks and obstacles obtained from the virtual environment, are passed into the network, which using 32, 7 x 7 x 4 (filter width, height and depth

respectively) dimensional filter convolved to produce hidden layer (Conv 1) of dimension 20 x 20 x 4, which is further convolved to produce a 9 x 9 x 64 - dimensional hidden layer (Conv 2) and then 7 x 7 x 64 - dimensional hidden layer (Conv 3). Next, the last hidden layer (Conv 3) is fully connected to 3136 neurons (FC5), further it is fully connected to 512 neurons (FC6) and lastly connected to 4 outputs, or search directions or actions – up, down, left and right. As the state input (current situation) is passed through the network, the network performs a forward propagation to predict a state-action value $Q(S_t, a)$ for each action and chooses an action based on the exploration rate.

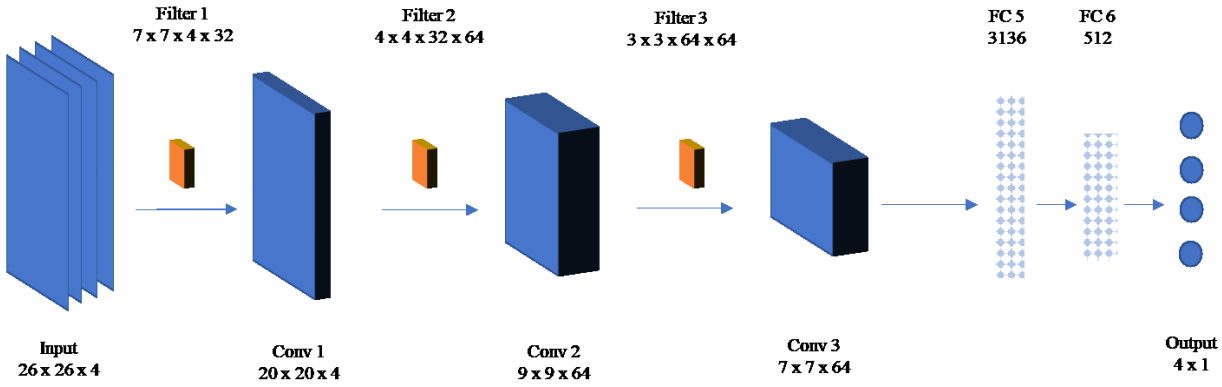


Figure 4: Deep convolutional neural network architecture used to determine action values.

A backward propagation process is performed to adjust the weights, which improves the state-action-value prediction. A loss function that compares the prediction $Q(S_t, a)$ and ground truth Y_t is employed to compute the partial derivative during the backward propagation process,

$$\delta = (Q(S_t, a) - Y_t)^2.$$

The function used for predicting ground truth state-action values for our model is (Van 2016),

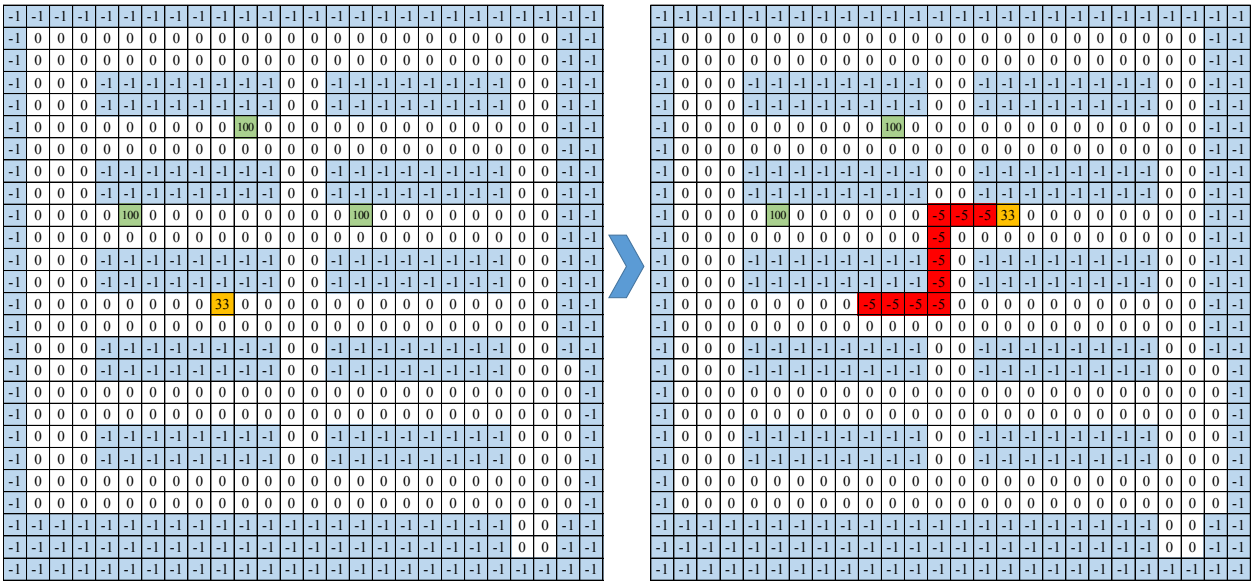
$$Y_t \equiv R_{t+1} + \gamma Q\left(S_{t+1}, \underset{a}{\operatorname{argmax}}\, Q(S_{t+1}, a;\ \theta_t), \theta_t^-\right),$$

where R_{t+1} is the reward from the system at time $t+1$, γ is the discount factor which determines the extent of future rewards allowed to influence current actions, $Q\ (S_{t+1}, a; \theta_t)$ is the Q state-action value based on a forward propagation given weights θ_t, and lastly θ_t^- is the target network weight. The target network is obtained by copying the current network after each 10,000 action steps. The forward and backward propagation processes are reviewed by Goodfellow et al. (2016).

The state input from the simulation environment is loaded into the virtual environment, which the agent uses for searching the shortest path task. The loaded environment is shown in Figure 5. The future state inputs from the virtual environment changes based on the search directions performed. The following numbers were used in the virtual environment to represent the various state transitions: -1 indicating obstacles (walls, shelves, etc.), 33 for the agent's current location and 100 for task locations. Further, to improve the agent's learning process we have used -5 to indicate previously searched or traversed locations of the agent.

Lastly, the experimental configurations are set to train and evaluate the agent's accuracy in determining the shortest path when there is two, three, and four tasks, keeping network design and hyper-parameters constant. Each scenario is run for 80,000 replications, where each replication terminates if the agent finds

a task location or has exceeded 250 action or search steps. We have observed that the limit on the maximum number of action steps per replication is important as it can accelerate or decelerate learning during the exploration stage. Based on the preliminary results on the average number of actions for a single task scenario, we observed values of less than 250 in all replications. Further, rewards system of the virtual environment is designed to penalize by -0.01 for every action or search step taken, -0.05 for revisiting a previously searched location, +1 for the step that reaches the task location and -1 for entering an obstacle.



(a) (b)

Figure 5: State representations - (a) Initial state input; and (b) Final state.

5 RESULTS

The performance of the autonomous vehicle/agent is measured by the accuracy of determining the closest task location with respect to the vehicle. The closest task can be determined by computing the distance from the vehicle to each goal location using the A* algorithm. If the goal location that autonomous agent chooses is the same as the closest task verified by the A*, the replication is marked as a success. If the autonomous agent fails to identify the closest task, we keep track of the difference between the distance to closest task and the task selected by the autonomous agent. For each 100 replications, the percentage success rate is collected, as well as the percentage of choosing a task that is 5 ft., 10 ft., and 15 ft. further than the closest task. That is, from a practical standpoint, we are interested in whether the vehicle made a good selection or a poor selection. We would consider a selecting an task that is within a few feet of the closest task to be acceptable. Figure 6, 7, 8 show the accuracy of the autonomous agent when there are two, three, and four tasks in the system, respectively. The dashed line and solid line sections represent the performance of the autonomous agent during the training and testing, respectively. Table 1 shows a comparison of the scenarios based on the average percentage and the standard deviation of the percentage of tasks selected that were within the specified distance of the closest task.

Note: The layout in our experimental setup is inspired from an actual warehouse modified to generalize to a typical warehouse space, where aisles are long, and vehicle paths are rectilinear. The author acknowledges that duration of training might be different for different layouts and different deep reinforcement learning network architectures.

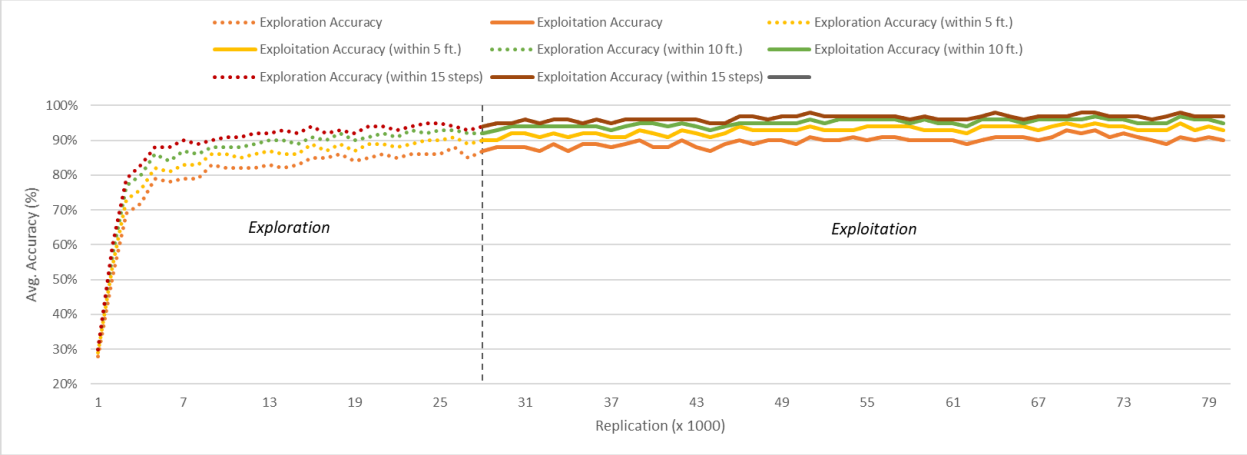


Figure 6: Performance of autonomous agent in two task scenario under exploration and exploitation.

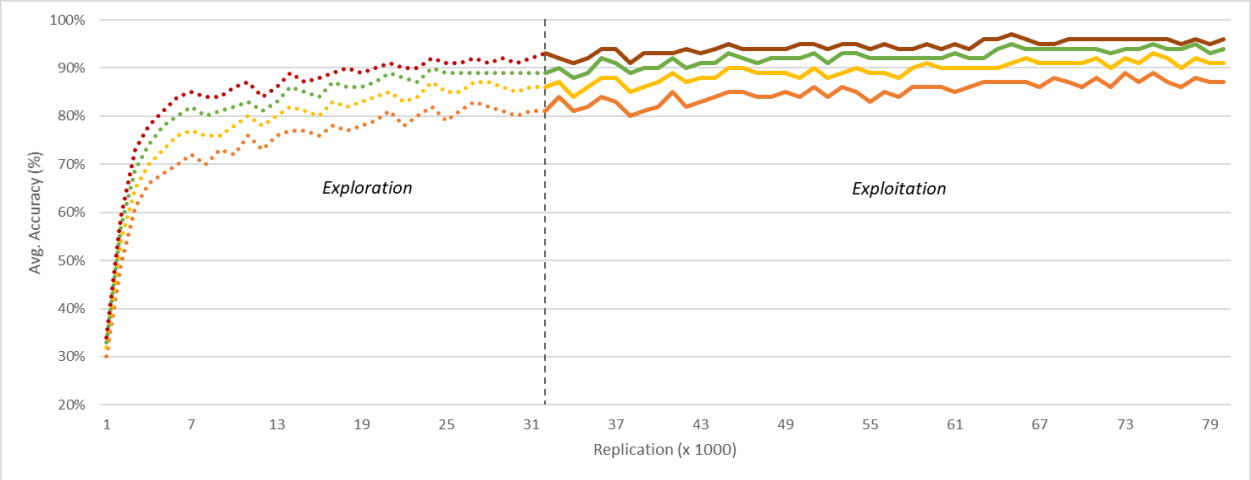


Figure 7: Performance of autonomous agent in three task scenario under exploration and exploitation.

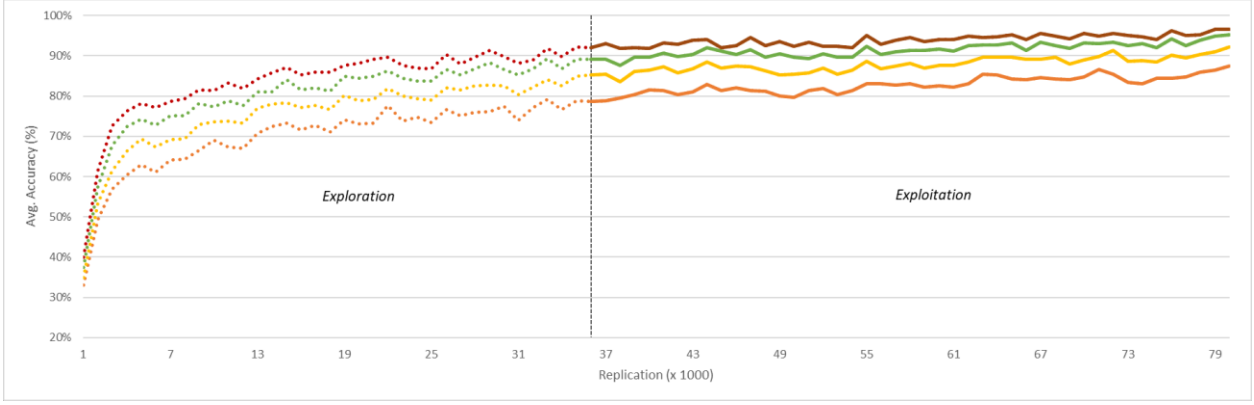


Figure 8: Performance of autonomous agent in four task scenario under exploration and exploitation.

Table 1. Comparison scenarios having two, three, and four tasks, respectively. Average percentage (std. dev.) of tasks selected that are within the specified distance of the closest task.

Accuracy (Closest Task)	2 Tasks	3 Tasks	4 Tasks
Within 0 ft.	89.76 (1.44)	85.11 (2.17)	82.82 (2.16)
Within 5 ft.	92.89 (1.22)	89.37 (1.99)	87.87 (1.87)
Within 10 ft.	95.02 (1.07)	92.29 (1.72)	91.51 (1.70)
Within 15 ft.	96.46 (0.92)	94.54 (1.38)	93.95 (1.32)

6 CONCLUSION

In conclusion, we have presented a deep reinforcement learning methodology for task selection by an autonomous vehicle in a warehouse environment. The results of the simulation-based experiment indicate that the method can consistently select the closest task given a random set of task location at a high level of performance. Based on these results, we anticipate that we will be able to extend this methodology the task selection of multiple autonomous vehicles where task priorities may be based on other factors such as due date, urgency, or other priority. Our future work includes this extension along with application of the methodology to a physical material handling system. Future work includes an evaluation of the robustness of our architecture on different layouts, and the proposed multiple autonomous vehicle extension along with application of the methodology to a physical material handling system.

ACKNOWLEDGMENTS

This research is sponsored in part by a grant form Toyota Material Handling of North America.

REFERENCES

Contreras-Cruz, M. A., V. Ayala-Ramirez, and U. H. Hernandez-Belmonte. 2015. "Mobile Robot Path Planning Using Artificial Bee Colony and Evolutionary Programming". *Applied Soft Computing* 30(C): 319-328.

Egbelu, P. J. and J. M. Tanchoco. 1984. "Characterization of Automatic Guided Vehicle Dispatching Rules". *International Journal of Production Research* 22(3):359-374.

Fang, Q. and C. Xie. 2004. "A Study on Intelligent Path Following and Control for Vision-based Automated Guided Vehicle". *Fifth World Congress on Intelligent Control and Automation,* 4811-4815. Piscataway, New Jersey: IEEE..

Fazlollahtabar, H. and M. Saidi-Mehrabad. 2015. "Methodologies to Optimize Automated Guided Vehicle Scheduling and Routing Problems: A Review Study". *Journal of Intelligent & Robotic Systems.* 77(3-4):525-545.

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio. 2016. *Deep learning*. Cambridge: MIT Press.

Hart, P.E., N. J. Nilsson and B. Raphael. 1968. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". *IEEE transactions on Systems Science and Cybernetics* 4(2):100-107.

Herrero-Perez, D. and H. Martinez-Barbera. 2010. "Modeling Distributed Transportation Systems Composed Of Flexible Automated Guided Vehicles in Flexible Manufacturing Systems". *IEEE Transactions on Industrial Informatics* 6(2):166-180.

Jeon, S. M., K. H. Kim, and H. Kopfer, 2011. "Routing Automated Guided Vehicles In Container Terminals Through The Q-Learning Technique". *Logistics Research* 3(1):19-27.

Le-Anh, T. and M. B. M. De Koster. 2006. "A Review of Design and Control of Automated Guided Vehicle Systems". *European Journal of Operational Research* 171(1):1-23.

Martínez-Barberá, H. and D. Herrero-Pérez. 2010. "Autonomous Navigation of an Automated Guided Vehicle in Industrial Environments". *Robotics and Computer-Integrated Manufacturing* 26(4):296-311.

Mnih, V., K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, M. 2013. "Playing Atari with Deep Reinforcement Learning". NIPS Deep Learning Workshop (arXiv:1312.5602) https://arxiv.org/abs/1312.5602?context=cs.LG, accessed 6 August 2018.

Mnih, V., K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, and S. Petersen. 2015. "Human-Level Control through Deep Reinforcement Learning". *Nature* 518(7540):529.

Ramer, C., J. Sessner, M. Scholz, X. Zhang, and J. Franke. 2015. "Fusing Low-Cost Sensor Data for Localization and Mapping of Automated Guided Vehicle Fleets in Indoor Applications". In *Proceedings of the* 2015 *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 65-70. Piscataway, New Jersey: IEEE.

Stentz, A. 1994. "Optimal and Efficient Path Planning For Partially-Known Environments". In *Proceedings of the IEEE International Conference on Robotics and Automation*, 3310-3317. Piscataway, New Jersey: IEEE.

Van Hasselt, H. 2010. "Double Q-learning." In *Proceedings of the 23rd Int. Conference on Neural Information Processing Systems*. 2613-2621. http://dl.acm.org/citation.cfm?id=2997046.2997187, accessed 6 August 2018.

Van Hasselt, H., A. Guez, and D. Silver. 2016. "Deep Reinforcement Learning with Double Q-Learning." In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, 2094-2100. http://dl.acm.org/citation.cfm?id=3016100.3016191, accessed 6 August 2018.

Veres, S. M., L. Molnar, N. K. Lincoln, and C. P. Morice. 2011. "Autonomous Vehicle Control Systems—A Review of Decision Making". *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering* 225(2):155-195.

AUTHOR BIOGRAPHIES

MAOJIA PATRICK LI is a Ph.D. student in Engineering at Rochester Instiutte of Techology. His e-mail address is mxl8487@rit.edu.

PRASHANT SANKARAN is a Master of Science (MS) student in Industrial and Systems Engineering at Rochester Institute of Technology. His research interests include the application of simulation, optimization, and machine learning approaches to supply chain and manufacturing systems. His e-mail address is pxs8917@rit.edu.

MICHAEL E. KUHL is a Professor in the Department of Industrial and Systems Engineering at Rochester Institute of Technology. He earned his Ph.D. in Industrial Engineering from North Carolina State University. His research interests include modeling and simulation of stochastic arrival processes, and the application of simulation and simulation-based optimization to systems including healthcare, manufacturing, cyber security, and project management. He is a member of the WSC Board of Director representing the INFORMS Simulation Society. He has also served WSC as Proceedings Editor (2005), Program Chair (2013), and Mobile App Chair (2014-2018). His e-mail address is Michael.Kuhl@rit.edu.

AMLAN GANGULY is an Associate Professor in the Computer Engineering Department at Rochester Institute of Technology. He earned his Ph.D. in Electrical and Computer Engineering from Washington

State University. His research interests are in energy-efficient interconnection architectures for multicore chips and multichip systems such as servers using novel technologies such as wireless and photonic interconnects and data center networks. His e-mail address is axgeec@rit.edu.

ANDRES KWASINSKI is a Professor in the Computer Engineering Department at Rochester Institute of Technology. He earned his Ph.D. degree in Electrical and Computer Engineering from the University of Maryland. He is Chief Editor for the IEEE SigPort and Area Editor for the IEEE Signal Processing Magazing. His research interests include cognitive radio and networking, multimedia communications and networking, Internet-of-Things, and smart infrastructures. Dr. Kwasinski is an IEEE Senior member. His e-mail address is axkeec@rit.edu.

RAYMOND PTUCHA is an Assistant Professor in the Computer Engineering Department at Rochester Institute of Technology. His research interests include machine learning, computer vision, robotics, graph processing and signal processing, all with an emphasis with deep learning. His the chair of the local IEEE Signal Processing Society and is passionate about STEM education. His e-mail address is rwpeec@rit.edu.