A MULTI-AGENT REINFORCEMENT LEARNING FRAMEWORK FOR INTELLIGENT MANUFACTURING WITH AUTONOMOUS MOBILE ROBOTS

Agrawal, Akash (1);
Won, Sung Jun (1);
Sharma, Tushar (2);
Deshpande, Mayuri (2);
McComb, Christopher (1)

1: The Pennsylvania State University;
2: Siemens Technology

ABSTRACT

Intelligent manufacturing (IM) embraces Industry 4.0 design principles to advance autonomy and increase manufacturing efficiency. However, many IM systems are created ad hoc, which limits the potential for generalizable design principles and operational guidelines. This work offers a standardizing framework for integrated job scheduling and navigation control in an autonomous mobile robot driven shop floor, an increasingly common IM paradigm. We specifically propose a multi-agent framework involving mobile robots, machines, humans. Like any cyberphysical system, the performance of IM systems is influenced by the construction of the underlying software platforms and the choice of the constituent algorithms. In this work, we demonstrate the use of reinforcement learning on a sub-system of the proposed framework and test its effectiveness in a dynamic scenario. The case study demonstrates collaboration amongst robots to maximize throughput and safety on the shop floor. Moreover, we observe nuanced behavior, including the ability to autonomously compensate for processing delays, and machine and robot failures in real time.

Keywords: Industry 4.0, Machine learning, Artificial intelligence, Intelligent Manufacturing Systems

Contact:
McComb, Christopher Carson
The Pennsylvania State University
School of Engineering Design, Technology, and Professional Programs
United States of America
mccomb@psu.edu

Cite this article: Agrawal, A., Won, S. J., Sharma, T., Deshpande, M., McComb, C. (2021) 'A Multi-Agent Reinforcement Learning Framework for Intelligent Manufacturing with Autonomous Mobile Robots', in *Proceedings of the International Conference on Engineering Design (ICED21),* Gothenburg, Sweden, 16-20 August 2021. DOI:10.1017/pds.2021.17

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

1 INTRODUCTION

Industry 4.0 is an information-intensive transformation of manufacturing and other fields driven by the convergence of technologies like cyber-physical systems, the internet of things, advanced robotics, and artificial intelligence. Decentralization, real-time capability, interoperability, service orientation, modularity, and virtualization have been identified as key design principles for implementing Industry 4.0 solutions (Hermann et al., 2016). Embracing these principles can lead to Intelligent Manufacturing (IM) systems with increased connectivity, enhanced autonomy, better customization, higher productivity, and more comprehensive monitoring (Wichmann et al., 2019). However, many modern IM systems are created in an ad hoc or bespoke manner, which limits the potential for generalizable design principles and operational guidelines. To promote the standardization of IM systems, various organizations have proposed frameworks ranging from high-level enterprise integration to low-level shop floor control (Li et al., 2018). This work proposes a standardizing framework for the integrated job scheduling and navigation control of an autonomous mobile robot driven shop floor, a promising and increasingly common IM paradigm.

The construction of software platforms and the choice of algorithms plays an important role in determining the performance of an IM system (Zhou and Le Cardinal, 2019). With advancements in machine vision, language, cloud infrastructures and deep learning, artificial intelligence platforms are increasingly useful in IM systems (Sharp et al., 2018; Wang et al., 2018; Wuest et al., 2016). Specifically, Reinforcement Learning (RL) offers adaptable, yet robust, control algorithms without the need to engineer specific system behaviours in robotic applications (Kober et al., 2013). This work further demonstrates the effectiveness of RL within the proposed standardizing framework in a simulated environment.

The framework introduced in this work assumes a multi-agent system. Multi-agent systems are commonly used to model enterprise integration and collaboration, process planning, job scheduling, and shop floor control in IM systems (Monostori et al., 2006; Shen et al., 2006). They benefit from several key inherent advantages, including flexibility, modularity, reconfigurability, and adaptability that align with the design principles for implementing Industry 4.0 solutions (Leusin et al., 2018). However, these advantages are constrained by challenges in agent organization, agent coordination, and negotiation, as well as the lack of relevant standards (Shen et al., 2006). With IM systems modelled as multi-agent systems, deep RL is a promising candidate to overcome several of the above challenges (Hernandez-Leal et al., 2019; Nguyen et al., 2020).

In an intelligent manufacturing system, software agents can represent a variety of entities, including decision-making software, human operators, and autonomous machines. Multi-agent systems for job scheduling aim to complete parallel and sequential jobs with limited manufacturing resources through effective shop floor control. For an autonomous mobile robot driven shop floor, job scheduling and navigation control of freely moving robots are closely coupled as the assignment of a job to a specific machine involves a robot navigating to that machine and placing material for processing. For a shop floor with multiple machines and robots, a centralized framework can lead to high computational complexity and is prone to deadlocks triggered by a single point of failure. With distributed computing and autonomy, a multi-agent system can give better scalability and local decision-making capabilities for dynamic changes like rush jobs and processing time delays, and failures like machine breakdowns and robot malfunctions. Moreover, enabling communication amongst the agents can promote better cooperation amongst the robots.

The research methodology adopted in the paper is as follows. In Section 2, we identify common characteristics of autonomous mobile robot driven shop floors that align with the Industry 4.0 design principles of decentralization, real-time capability, and interoperability as well as the primary objectives of such a shop floor. Furthermore, we identify gaps in existing IM frameworks and identify RL as a promising candidate for the navigation of mobile robots and job scheduling. In Section 3, we propose a framework based on agent technology that addresses gaps identified in Section 2. We further translate this framework into a mathematical state-space representation of the system for use in an RL algorithm. In Section 4, we showcase the effectiveness of the framework and state-space representation by applying them to a simple material handling case study. In this case study, we observe that the agents control the navigation of the mobile robots to collaboratively schedule jobs at the machines of the shop floor. Moreover, we validate the robustness of the learned control policy with a test involving processing time delays, a machine breakdown, and a robot malfunction.

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

2 BACKGROUND

2.1 Multi Agent Intelligent Manufacturing Systems

Cyber-physical systems (Lee et al., 2015; Wang et al., 2015) are the building blocks of intelligent manufacturing, with the cyber portion consisting of software entities that handle data acquisition, data processing, and data interpretation to undertake intelligent, real-time and adaptive decision-making. These entities communicate with each other to compose the multi-agent system. The physical portion consists of physical industrial assets like manipulator robots, mobile robots, machines, conveyors, fixtures, tools, and inspection devices. In agent technology (Jennings and Wooldridge, 1998), it is common to specify an agent in terms of two components: perception and action. Perception is accomplished through sensors embedded in the physical portion of the system. Action is accomplished when the cyber portion of the system controls the physical portion through embedded equipment like actuators. These cyber-physical systems work towards meeting various objectives involving throughput, equipment utilization, energy consumption, maintenance cost, and safety. This work focuses on a shop floor with machines, autonomous mobile robots, and human workers engaged in processing, transporting, and monitoring activities to meet the objectives of maximizing throughput and safety.

Autonomous mobile robots may improve the efficiency, adaptability, and flexibility of intelligent manufacturing systems as compared to traditional transport systems like conveyors (Fragapane et al., 2020). They can move freely using sensors that detect static and dynamic assets in the shop floor. This enables them to undertake robotic material handling, decentralized material transport to schedule jobs at various machines that process this material. Congestion and material queues are often avoided, leading to an improvement in system throughput. They can also communicate and work in conjunction with automation lines, machines, manipulator robots and human workers. In doing so they can respond to dynamic changes and failures in the shop floor like rush jobs, processing time delays, machine breakdowns, or robot malfunctions. Dynamic path planning and responsiveness to failure are essential characteristics of such shop floors that relate to the Industry 4.0 design principles of decentralization and real-time capability (Hermann et al., 2016). In such a system, all cyber-physical assets should be able to communicate with each other through standard protocols. This enables interoperability (Hermann et al., 2016) between systems of various manufacturers, a key Industry 4.0 design principle.

The collaboration and interaction of human workers with these cyber physical systems is also crucial for the success of intelligent manufacturing (Cimini et al., 2020; Zhou et al., 2019). They collaborate with robots to monitor the activities on the shop floor to report anomalies, thereby contributing to the system's real-time capability (Hermann et al., 2016). Their safety alongside robots is an essential objective for such shop floors. The robots should change their behaviour based on the somewhat stochastic nature of human workers' activity in the environment. Thus, we identify dynamic path planning, responsiveness to failure, standardized communication, and collaboration with humans as key characteristics of autonomous mobile robot driven shop floors that translate from the Industry 4.0 design principles. The integrated job scheduling and navigation control of mobile robots to meet the objectives of throughput and safety in this context is a challenging problem in IM systems.

Extensive research has been conducted to implement multi-agent systems for job scheduling and shop floor control in intelligent manufacturing (Leitão, 2009; Monostori et al., 2006; Parente et al., 2020; Shen et al., 2006). However, there has been limited work that has used a multi-agent system to encapsulate all key characteristics of autonomous mobile robot driven shop floors. Malus et al. (2020) have proposed a framework where mobile robots are represented by agents that bid on individual transport jobs based on the start location of the current job, the location of the mobile robot, the number of jobs assigned and the end location of the last assigned job. However, it does not allow the change of a job from one robot to another once a bid has been made. During dynamic changes and failures, it could be beneficial to reassign the order amongst the robots based on their locations, orders, and the status of jobs at the machines. Moreover, a multi-agent framework of robots that facilitates a truly dynamic nature in job scheduling should enable the robots to assign jobs by navigating to the machines in a decentralized and real-time manner.

2.2 Reinforcement Learning For Navigation And Job Scheduling

Reinforcement learning (Sutton and Barto, 2018; Watkins and Dayan, 1992; Williams, 1992) is a class of machine learning that enables an agent or multi-agent system to learn a policy in an interactive

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

environment by trial and error using feedback from its own actions and experiences. Its success was truly marked when Mnih et al. (2015) made use of a deep Q network in creating an agent that outperformed a professional player in a series of Atari games. Moreover, the reinforcement learning based AlphaGo program was the first program that could beat the best professional players in Go, the most challenging of classic games for artificial intelligence (Silver et al., 2016). As of now it has found applications in various domains including recommender systems, computer systems, energy, finance, healthcare, robotics, and transportation (Li, 2019).

RL is typically used when a problem can be framed as a Markov decision process. A Markov decision process consists of a set of finite environment states, a set of possible actions in each state, a transition model, and a reward function. At each learning step, the agent senses the environment, takes an action, and transits to a new state in the environment. The quality of each transition is evaluated by the reward function. The objective of the agent is to maximize this reward. In model-based methods, the agent learns to model the environment from its observations and then plans a solution using that model. On the other hand, model free methods depend on sampling and simulation to estimate rewards without knowing the inner working of the model.

A robot's ability to determine its own position in its reference frame and then to plan a path towards some target location by avoiding dangers such as collisions is referred to as robot navigation. RL is being extensively used for the navigation of ground robots by mapping raw sensor measurements to its navigation commands for obstacle avoidance (Fan et al., 2020). For instance, Han et al. (2020) trained a homogenous multi-agent system of navigating ground robots that used proximal policy optimization to maximize target and obstacle avoidance rewards based on their poses and velocities. This approach is suitable for the navigation control of mobile robots to various machines on a manufacturing shop floor. However, using a centralized approach for allocating targets limits its applicability for job scheduling. In other work, a multi-agent deep deterministic policy gradient algorithm has shown promise for the integrated target assignment and path planning of aerial vehicles (Qie et al., 2019). Multi-agent RL has also been used to train aerial vehicles to cooperatively perform field coverage making it suitable for surveillance in intelligent manufacturing shop floors (Pham et al., 2018). Further, deep RL has enabled unmanned ground and aerial vehicles to form a coalition that is complementary and cooperative for completing tasks that they are incapable of achieving alone (Zhang et al., 2020).

Deep RL is also being increasingly used for job scheduling in intelligent manufacturing. A proximal policy optimization algorithm was employed by formulating the order batching and sequencing problem of a warehouse as a semi-Markov decision process that performs better than heuristic approaches (Cals et al., 2020). A trust region policy optimization based adaptive algorithm has been shown to optimize the key performance indicators and lead time on orders in manufacturing (Kuhnle et al., 2019). Malus et al. (2020) focussed on an autonomous mobile robot shop floor with a simple layout wherein the multi-agent learning problem was modelled as a Markov decision process. They use a twin delayed deep deterministic algorithm where state transitions and rewards only occur during job assignment. This limited the granularity of the actions that the agents could learn. Moreover, this also kept the agent from learning how to address dynamic scenarios which may require changing the job while a robot is already navigating to an assigned job.

3 MULTI-AGENT FRAMEWORK FOR MOBILE ROBOT DRIVEN SHOP FLOOR

We propose a multi-agent framework for an autonomous mobile robot driven shop floor, wherein the sensory, control, and communication protocols have been translated from the Industry 4.0 design principles. These aspects include dynamic path planning, responsiveness to failure, standardized communication, and collaboration with humans. In this framework, machines and robots are represented by agents that partially perceive and autonomously act upon the shop floor environment through embedded sensors and controllers. Human workers are represented as operator agents that also perceive the shop floor and act based on their own intelligence. Agents coordinate implicitly through the perception of other agents' actions within the environment. Moreover, these agents also explicitly communicate with each other through the communication server agent with one-way or two-way communication channels. A schematic of the proposed framework is shown in Figure 1.

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

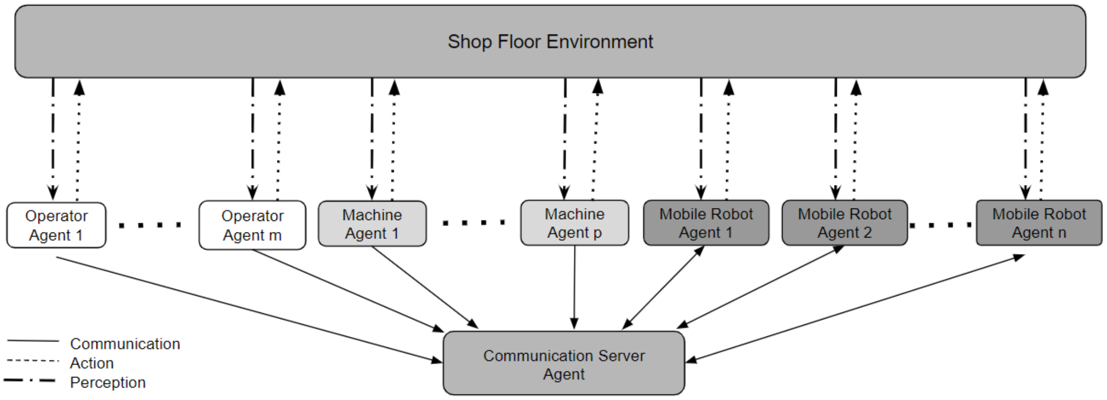

Figure 1. Multi-agent systems framework

In Figure 1, the machine agents represent the cyber portion of any stationary machine that processes raw material into a finished or semi-finished part (e.g., CNC machine). We assume that a machine can hold and process only one part at a time. This agent stores the status of the machines. The possible states are idle to receive material, completed processing, processing, or breakdown. It can perceive the presence or absence of raw material or a part when a robot drops or picks the same, which triggers its actions and changes the status accordingly. This agent communicates the status of the machine to the communication server agent.

The mobile robot agents represent the cyber portion of any mobile robot that transports and handles material in the shop floor. These robots may be embodied as ground robots, drones, or other types. We assume that the robots have the carrying capacity for the raw material and parts that can last for a finite number of processing cycles. This agent stores the position and velocity of the robot. It has a two-way communication channel with the server as shown in Figure 1. It communicates the position and velocity to the server and receives the position and velocity of other robots and the status of all machines to respond to dynamic changes and failures. It can perceive other objects within a set distance from the robot on the shop floor using proximity or visual sensors. The action of this agent is to send signals to the embedded controller on the robot to navigate on the shop floor and pick or place material at the machines. Based on the position and velocity of the robot, its sensor inputs, position and velocity of other robots and status of machines, it can dynamically replan its path. Additionally, if it perceives any anomalies on the shop floor (e.g., a machine stuck in the processing state) it communicates that observation to the server. In this case, the server can override the status as breakdown before communicating to other agents so that they can prioritize other machines.

As noted above, the operator agents represent human workers that report anomalies on the shop floor. Moreover, they can also do quality checks at the machines and communicate to the server agent if a machine status needs to be overridden to the breakdown state rather than getting into the idle state for more jobs. In terms of navigation, the position and velocity space of these agents is limited to the shop floor unlike some mobile robots that can fly. It is interesting to note that the navigation behaviour of humans is stochastic in nature and can pose a challenge for agents that control the robots around humans.

Finally, the communication server agent provides marginal centralized control and enables all agents to communicate through a standard protocol. Moreover, it can override the machine status to indicate a breakdown based on the surveillance inputs from the robots and human workers.

In order to permit the use of RL to guide the mobile robots in this system, we define its state-space (S) as the combination of the machine states (S_1), mobile robot states (S_2) and human states (S_3):

$$S = \{S_1, S_2, S_3\} \tag{1}$$

$$S_1 = \{\{I, C, P, B\}_1, \dots \{I, C, P, B\}_p\} \tag{2}$$

$$S_2 = \{\{x, y, z, v_x, v_y, v_z\}_1, \dots \{x, y, z, v_x, v_y, v_z\}_n; x, y, z \in Q\} \tag{3}$$

$$S_3 = \{\{X, Y, V_X, V_Y\}_1, \dots \{X, Y, V_X, V_Y\}_m; X, Y \in R\} \tag{4}$$

where p is the number of machines, I, C, P and B are the idle, completed, processing and breakdown statuses, x, y, z, and v_x, v_y, v_z are the positions and velocities of robots, n is the number of robots, Q is the feasible region for the movement of robots, X and Y, and V_X and V_Y are the positions and velocities of the humans, m is the number of humans, and R is the feasible region for the movement of humans.

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

4 A REINFORCEMENT LEARNING CASE STUDY

4.1 Case Study Description

The motivation for the case study is to demonstrate the effectiveness of a deep reinforcement learning algorithm on the proposed multi-agent systems framework. The learning problem is modelled as a Markov decision process with communication amongst homogenous agents sharing a common policy. A proximal policy optimization algorithm with curiosity driven exploration (Pathak et al., 2017; Schulman et al., 2017) is used for training the policy. Reinforcement learning typically requires a very high volume of trial-and-error episodes to learn a good policy. Therefore, simulators are required to achieve results in a cost-effective, timely and safe manner. Here, we use Unity, a game engine that is garnering interest in both industry and academia for engineering applications like virtual product dissection, virtual assembly, and machining, visualizing products through augmented or virtual reality, building navigation systems, and simulating agent-based models. We make use of Unity's built in ML Agents Toolkit for our reinforcement learning algorithm (Juliani et al., 2018), but this work can be translated to other simulators.

The Unity based virtual environment is shown in Figure 2 (a). For this case study, we model three ground robots, seven machines and two human workers. All robots are ground robots with discrete orthogonal motion at a fixed speed ($\mathrm{v_z} = 0, \mathrm{v_x} \cdot \mathrm{v_y} = 0, |\mathrm{v}| \in \{0, k_{speed}\}\ in\ S_2$) and are responsible for material handling. For reasons of safety, it is common for the shop floor to have barriers and guidelines to guide the movement of humans and robots. Each machine converts raw material to different finished parts, and this processing can be subject to processing time delays. In this case, the robots essentially need to pick up the finished part after processing and place raw material for the next cycle of the machine. This enables the demonstration of the robots' ability to replan its path to schedule jobs at various locations on the shop floor with dynamic changes. As a robot always places new raw material when it picks a finished part from a machine, there remain only two machine states, completed (C) and processing (P). We further assume that this exchange happens instantaneously when the robot reaches a machine. The robot also needs to ensure that it avoids collisions with other assets on the shop floor while navigating. Each human worker oversees three machines, while one of the machines does not require oversight. Human agents navigate using the shortest path without crossing the guidelines from machine to machine.

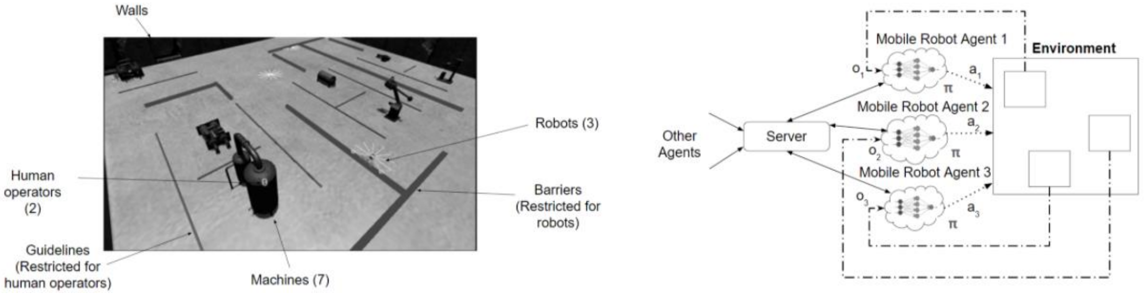

Figure 2. (a) Virtual environment of the shop floor (b) Multi-agent training scenario

Training via RL is focused on the three homogenous mobile robot agents. The learning problem is formulated as a Markov decision process with all agents having a common policy network, π as shown in Figure 2 (b). The observation, action, and rewards for the same are defined in short below:

- Observations (o_i): Robot position and velocity as in S_2, perception sensor observations (transforms of positions in S_1 and S_2 and positions of static assets), position and velocity of other robots as in S_2 (via communication), machine statuses (C or P) as in S_1 (via communication)
- Actions (a_i): Discrete movement ($\mathrm{v_z} = 0, \mathrm{v_x} \cdot \mathrm{v_y} = 0, |\mathrm{v}| \in \{0, \mathrm{k_{speed}}\}\ in\ S_2$)
- Rewards: Positive reward per machine job to meet the throughput objective. Reward increases for subsequent machines to encourage the agent. Negative reward for movement actions to encourage navigation by shortest path to contribute to throughput. Negative reward for collision with wall or barrier, humans, other robots and going to machines that are processing to meet the safety objective.

We perform the training with one processing cycle per machine and expect each agent to maximize the number of jobs that it completes, thereby maximizing its reward. In this sense, the agents are in a state

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

of competition for limited reward resources. However, we penalize the movement of the robot and impose a constraint on the episode length. Thereby, it is expected that over several episodes of training the agents will learn to cooperate with each other to complete jobs with the least travel distance and within the episode limit. Specifically, they are expected to learn that their individual rewards can be maximized by choosing machines in a distributed way based on the location of the robots and the status of the machines. Accordingly, the problem is a mixed cooperative-competitive problem. We also expect the agents to ensure collision avoidance with other assets based on its sensor inputs. For building a robust policy, the robots and humans are spawned at random locations on the shop floor in each episode. This also exposes the agents within an episode to varying machine states based on where the robots are initially spawned. Thereby, we validated the learned policy over repeated cycles involving processing time delays. a machine breakdown and a robot malfunction. The agents can essentially command the scheduling of the jobs and the navigation of the robots in an integrated and dynamic manner to meet the shop floor objectives of maximizing throughput and safety.

4.2 Case Study Results

The results of training the agents using a proximal policy optimization with curiosity driven exploration are shown in Figure 3. We observe that the cumulative reward increases over time and reaches the expected value. The episode length is the mean length of each episode in the environment for all agents. For the initial few episodes, the agents struggle to complete the tasks within the episode limit, but we observe a drop in the episode length as the agents learn. The policy loss is the mean magnitude of the policy loss function and correlates to how much the policy is changing. As expected, these values oscillate with a decreasing magnitude which is less than one. The value loss is the mean loss of the value function update and correlates to how well the model can predict the value of each state. As expected, the values increase while the agent is learning. Entropy indicates the randomness of the decisions being made and should slowly decrease during a successful training process. This is observed in the case study. The learning rate determines the size of the step as the algorithm searches for the optimal policy. It shows a linear decrease with time as specified for training.

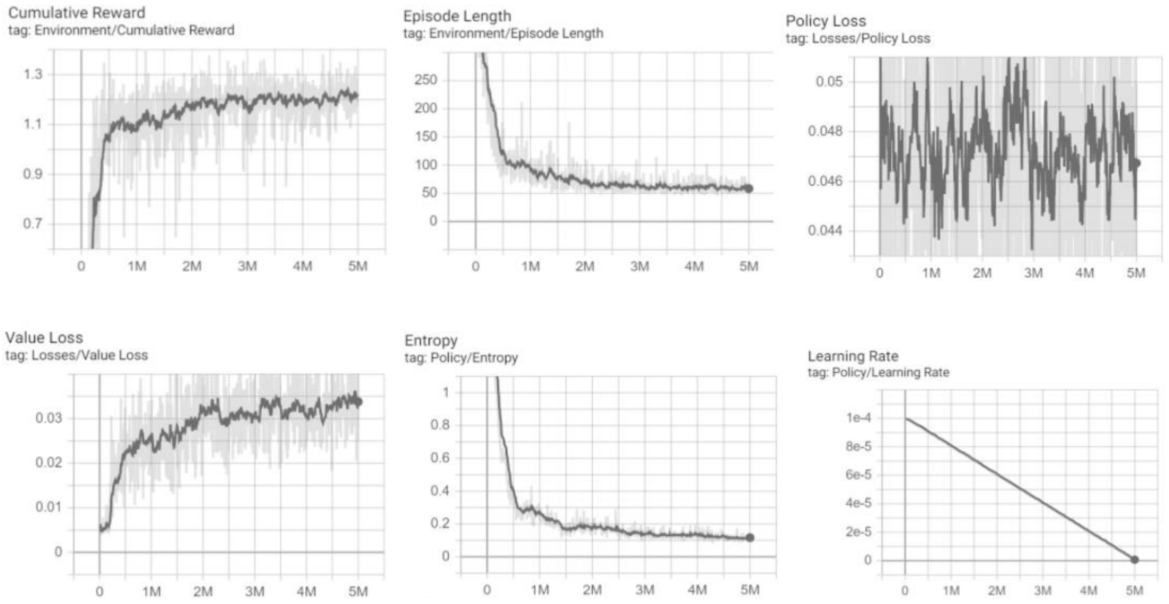

Figure 3. Training statistics

We now discuss some observations from training the policy. Figure 4 (a) shows an episode where the three robots collaboratively schedule jobs at all the machines within the episode limit. The robots are navigating along a path that is close to the shortest orthogonal path around the barriers to reach the machines. Thereby, the robots learn to maximize the throughput of the shop floor. However, as the agents are exposed to only one machine cycle per episode, they do not learn that in certain scenarios it would be more rewarding to halt at a location near a machine to schedule the next job than to compete with another robot for a machine that is far away. Lastly, the robots meet the safety objective by avoiding collisions with the barriers, walls, and machines but are exposed to humans very few times.

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

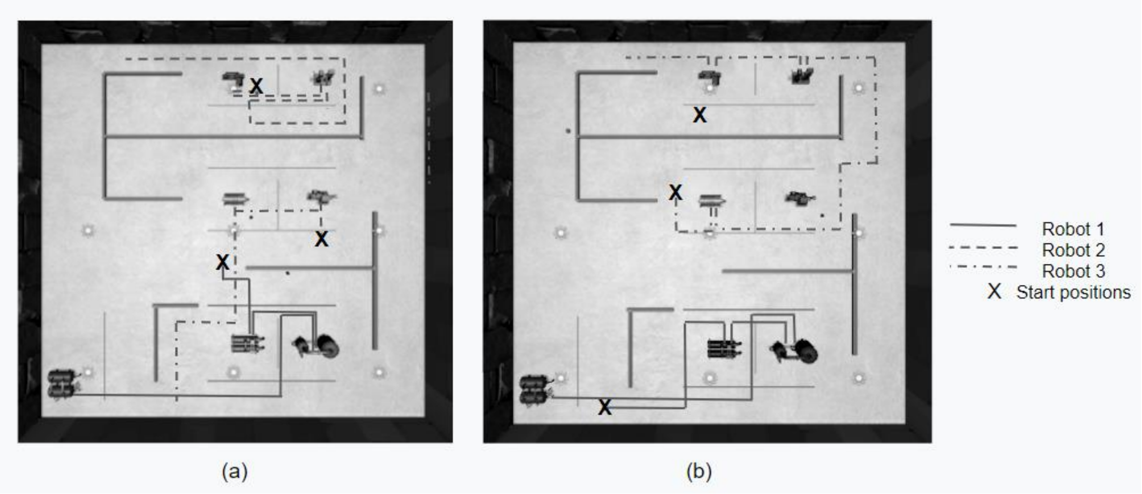

Figure 4. Robot navigation paths

To ensure the robustness of the policy in dynamic scenarios and failures, we perform a validation test wherein the machines can have processing delays of up to 50% over repeated cycles. Moreover, we introduce two failures in the form of a machine breakdown and robot malfunction after a few cycles. Specifically, machine 5 and robot 2 get stuck in the processing and halt states, respectively. Figure 4 (b) shows the robot paths for a short portion of this study post both the failures. Figure 5 shows the machine schedules of the entire study. The x-axis represents time, the y-axis represents the states of the machines (0 for completed, 1 for processing) and the identification number of the robot that schedules the job at the machine appears on the top of each plot whenever the machine starts the job. Firstly, we observe that the robots can schedule jobs at the machines with varying processing times. However, there is potential for improving the throughput by training over repeated cycles with different complexities. Secondly, the policy can adapt to the failure of machine 5 as the robots continue to schedule jobs at other machines. Lastly, the policy also adapts to the failure of a robot midway of the test. Consequently, there is also an increase in delays for scheduling jobs at the machines after this failure as the remaining robots need to travel longer distances to reach all the machines.

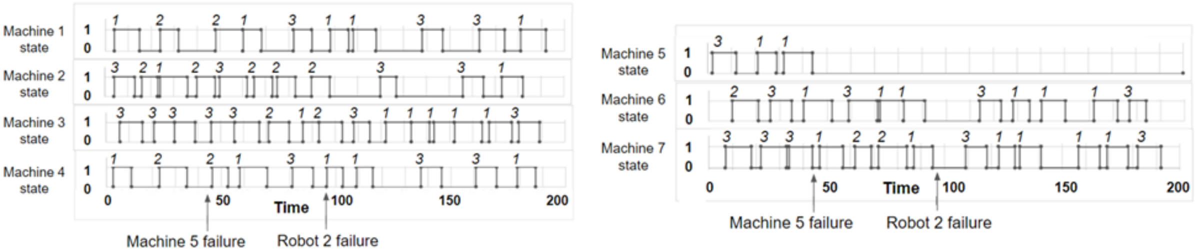

Figure 5. Machine schedules for the dynamic test

The reader is referred to the GitHub repository for the full code used in this work as well as videos of several training and testing episodes (https://github.com/THREDgroup/multi-agent-robots).

5 CONCLUSION

This work proposes a multi-agent framework for the integrated job scheduling and navigation control of a system involving autonomous mobile robots, a promising and increasingly common IM paradigm. Specifically, the framework encapsulates the characteristics of dynamic path planning, responsiveness to failure, standardized communication, and collaboration with humans. These translate from the Industry 4.0 design principles of decentralization, real-time capability, and interoperability. We also demonstrate the effectiveness of the framework through a case study, wherein a proximal policy algorithm is used for training a system of collaborative mobile robots to perform a material handling task with maximum throughput and safety. We validate the robustness of the learned policy in a scenario with processing delays, and a machine and robot failure.

Future work should explore the extension of this framework to IM systems with more diverse industrial assets. This opens the possibility for embodying additional Industry 4.0 design principles like service orientation, modularity, and virtualization. For instance, a web service around product and order agents can be composed to provide service orientation. Modularity can be incorporated by training with varying number of machines and robots or by using transfer learning. Lastly, simulated environments used in RL algorithms can be used in digital twin efforts for virtualization. A comprehensive standardizing IM framework will guide the design of constituent cyber-physical systems at the shop floor level, ultimately leading to easier integration and accessibility by higher levels of the system.

Future work should also pursue more robust validation by quantifying the objectives of throughput and safety and comparing against commonly used heuristics. In this effort, the learned policy can also be improved by training over repeated cycles with distinct system constraints and diverse anomalies. In this work, we observed that the mobile robots are infrequently exposed to safety-critical scenarios, such as navigating around humans. A more nuanced sampling strategy should be employed for exposing the agent to such scenarios. In addition, incorporating visual sensors with convolutional neural network-based approaches for object detection and long short-term memory networks for mapless navigation could lead to more robust navigation. Lastly, there is potential to explore different deep RL algorithms, policy network definitions, and training schemes.

REFERENCES

Cals, B., Zhang, Y., Dijkman, R. and van Dorst, C. (2020), “Solving the Order Batching and Sequencing Problem using Deep Reinforcement Learning”, ArXiv, pp. 1–31.

Cimini, C., Pirola, F., Pinto, R. and Cavalieri, S. (2020), “A human-in-the-loop manufacturing control architecture for the next generation of production systems”, Journal of Manufacturing Systems, Elsevier, Vol. 54 No. July 2019, pp. 258–271. https://doi.org/10.1016/j.jmsy.2020.01.002

Fan, T., Long, P., Liu, W. and Pan, J. (2020), “Distributed multi-robot collision avoidance via deep reinforcement learning for navigation in complex scenarios”, The International Journal of Robotics Research, Vol. 39 No. 7, pp. 856–892. https://doi.org/10.1177/0278364920916531

Fragapane, G., Ivanov, D., Peron, M., Sgarbossa, F. and Strandhagen, J.O. (2020), “Increasing flexibility and productivity in Industry 4.0 production networks with autonomous mobile robots and smart intralogistics”, Annals of Operations Research, Springer US. https://doi.org/10.1007/s10479-020-03526-7

Han, R., Chen, S. and Hao, Q. (2020), “Cooperative Multi-Robot Navigation in Dynamic Environment with Deep Reinforcement Learning”, 2020 IEEE International Conference on Robotics and Automation (ICRA), IEEE, pp. 448–454. https://doi.org/10.1109/icra40945.2020.9197209

Hermann, M., Pentek, T. and Otto, B. (2016), “Design Principles for Industrie 4.0 Scenarios”, 2016 49th Hawaii International Conference on System Sciences (HICSS), IEEE, pp. 3928–3937. https://doi.org/10.1109/hicss.2016.488

Hernandez-Leal, P., Kartal, B. and Taylor, M.E. (2019), “A survey and critique of multiagent deep reinforcement learning”, Autonomous Agents and Multi-Agent Systems, Springer US, Vol. 33 No. 6, pp. 750–797. https://doi.org/10.1007/s10458-019-09421-1

Jennings, N.R. and Wooldridge, M.J. (1998), Agent Technology: Foundations, Applications, and Markets, edited by Jennings, N.R. and Wooldridge, M.J., Springer-Verlag, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-03678-5

Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., et al. (2018), “Unity: A General Platform for Intelligent Agents”, pp. 1–28.

Kober, J., Bagnell, J.A. and Peters, J. (2013), “Reinforcement learning in robotics: A survey”, The International Journal of Robotics Research, Vol. 32 No. 11, pp. 1238–1274. https://doi.org/10.1177/0278364913495721

Kuhnle, A., Schäfer, L., Stricker, N. and Lanza, G. (2019), “Design, Implementation and Evaluation of Reinforcement Learning for an Adaptive Order Dispatching in Job Shop Manufacturing Systems”, Procedia CIRP, Elsevier B.V., Vol. 81, pp. 234–239. https://doi.org/10.1016/j.procir.2019.03.041

Lee, J., Bagheri, B. and Kao, H.-A. (2015), “A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems”, Manufacturing Letters, Society of Manufacturing Engineers (SME), Vol. 3, pp. 18–23. https://doi.org/10.1016/j.mfglet.2014.12.001

Leitão, P. (2009), “Agent-based distributed manufacturing control: A state-of-the-art survey”, Engineering Applications of Artificial Intelligence, Vol. 22 No. 7, pp. 979–991. https://doi.org/10.1016/j.engappai.2008.09.005

Leusin, M., Frazzon, E., Uriona Maldonado, M., Kück, M. and Freitag, M. (2018), “Solving the Job-Shop Scheduling Problem in the Industry 4.0 Era”, Technologies, Vol. 6 No. 4, p. 107. https://doi.org/10.3390/technologies6040107

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press

Li, Q., Tang, Q., Chan, I., Wei, H., Pu, Y., Jiang, H., Li, J., et al. (2018), “Smart manufacturing standardization: Architectures, reference models and standards framework”, Computers in Industry, Elsevier, Vol. 101 No. July, pp. 91–106. https://doi.org/10.1016/j.compind.2018.06.005

Li, Y. (2019), “Reinforcement Learning Applications”, ArXiv, pp. 1–41.

Malus, A., Kozjek, D. and Vrabič, R. (2020), “Real-time order dispatching for a fleet of autonomous mobile robots using multi-agent reinforcement learning”, CIRP Annals, Vol. 69 No. 1, pp. 397–400. https://doi.org/10.1016/j.cirp.2020.04.001

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J., Bellemare, M.G., Graves, A., et al. (2015), “Human-level control through deep reinforcement learning”, Nature, Vol. 518 No. 7540, pp. 529–533. https://doi.org/10.1038/nature14236

Monostori, L., Váncza, J. and Kumara, S.R.T. (2006), “Agent-Based Systems for Manufacturing”, CIRP Annals, Vol. 55 No. 2, pp. 697–720. https://doi.org/10.1016/j.cirp.2006.10.004

Nguyen, T.T., Nguyen, N.D. and Nahavandi, S. (2020), “Deep Reinforcement Learning for Multiagent Systems: A Review of Challenges, Solutions, and Applications”, IEEE Transactions on Cybernetics, Vol. 50 No. 9, pp. 3826–3839. https://doi.org/10.1109/tcyb.2020.2977374

Parente, M., Figueira, G., Amorim, P. and Marques, A. (2020), “Production scheduling in the context of Industry 4.0: review and trends”, International Journal of Production Research, Taylor & Francis, Vol. 58 No. 17, pp. 5401–5431. https://doi.org/10.1080/00207543.2020.1718794

Pathak, D., Agrawal, P., Efros, A.A. and Darrell, T. (2017), “Curiosity-driven Exploration by Self-supervised Prediction”, 34th International Conference on Machine Learning, ICML 2017, Vol. 6, pp. 4261–4270.

Pham, H.X., La, H.M., Feil-Seifer, D. and Nefian, A. (2018), “Cooperative and Distributed Reinforcement Learning of Drones for Field Coverage”, ArXiv, available at: http://arxiv.org/abs/1803.07250.

Qie, H., Shi, D., Shen, T., Xu, X., Li, Y. and Wang, L. (2019), “Joint Optimization of Multi-UAV Target Assignment and Path Planning Based on Multi-Agent Reinforcement Learning”, IEEE Access, IEEE, Vol. 7, pp. 146264–146272. https://doi.org/10.1109/access.2019.2943253

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O. (2017), “Proximal Policy Optimization Algorithms”, ArXiv, pp. 1–12.

Sharp, M., Ak, R. and Hedberg, T. (2018), “A survey of the advancing use and development of machine learning in smart manufacturing”, Journal of Manufacturing Systems, Vol. 48, pp. 170–179. https://doi.org/10.1016/j.jmsy.2018.02.004

Shen, W., Hao, Q., Yoon, H.J. and Norrie, D.H. (2006), “Applications of agent-based systems in intelligent manufacturing: An updated review”, Advanced Engineering Informatics, Vol. 20 No. 4, pp. 415–431. https://doi.org/10.1016/j.aei.2006.05.004

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., et al. (2016), “Mastering the game of Go with deep neural networks and tree search”, Nature, England, Vol. 529 No. 7587, pp. 484–489. https://doi.org/10.1038/nature16961

Sutton, R.S. and Barto, A.G. (2018), Reinforcement Learning: An Introduction, MIT press.

Wang, J., Ma, Y., Zhang, L., Gao, R.X. and Wu, D. (2018), “Deep learning for smart manufacturing: Methods and applications”, Journal of Manufacturing Systems, The Society of Manufacturing Engineers, Vol. 48, pp. 144–156. https://doi.org/10.1016/j.jmsy.2018.01.003

Wang, L., Törngren, M. and Onori, M. (2015), “Current status and advancement of cyber-physical systems in manufacturing”, Journal of Manufacturing Systems, Vol. 37 No. May, pp. 517–527. https://doi.org/10.1016/j.jmsy.2015.04.008

Watkins, C.J.C.H. and Dayan, P. (1992), “Q-learning”, Machine Learning, Vol. 8 No. 3–4, pp. 279–292. https://doi.org/10.1007/bf00992698

Wichmann, R.L., Eisenbart, B. and Gericke, K. (2019), “The Direction of Industry: A Literature Review on Industry 4.0”, Proceedings of the Design Society: International Conference on Engineering Design, Vol. 1 No. 1, pp. 2129–2138. https://doi.org/10.1017/dsi.2019.219

Williams, R.J. (1992), “Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning”, Machine Learning, Vol. 8 No. 3, pp. 229–256. https://doi.org/10.1007/bf00992696

Wuest, T., Weimer, D., Irgens, C. and Thoben, K.-D. (2016), “Machine learning in manufacturing: advantages, challenges, and applications”, Production & Manufacturing Research, Taylor & Francis, Vol. 4 No. 1, pp. 23–45. https://doi.org/10.1080/21693277.2016.1192517

Zhang, J., Yu, Z., Mao, S., Periaswamy, S.C.G., Patton, J. and Xia, X. (2020), “IADRL: Imitation Augmented Deep Reinforcement Learning Enabled UGV-UAV Coalition for Tasking in Complex Environments”, IEEE Access, Vol. 8, pp. 102335–102347. https://doi.org/10.1109/access.2020.2997304

Zhou, J., Zhou, Y., Wang, B. and Zang, J. (2019), “Human–Cyber–Physical Systems (HCPSs) in the Context of New-Generation Intelligent Manufacturing”, Engineering, Vol. 5 No. 4, pp. 624–636. https://doi.org/10.1016/j.eng.2019.07.015

Zhou, R. and Le Cardinal, J. (2019), “Exploring the Impacts of Industry 4.0 from a Macroscopic Perspective”, Proceedings of the Design Society: International Conference on Engineering Design, Vol. 1 No. 1, pp. 2111–2120. https://doi.org/10.1017/dsi.2019.217

https://doi.org/10.1017/pds.2021.17 Published online by Cambridge University Press