AGV Traffic Management: Simulation-Based Deadlock Resolution and Collision Reduction via Deep RL with PPO

Mustafa Jelibaghu[1], Michael Eley[1], Oliver Rose[2], Alexander Palatnik[1], Leon Roth[1], Tim Thorwart[1]

[1]University of Applied Sciences Aschaffenburg, Würzburger Straße 45, 63743 Aschaffenburg, Germany; *{mustafa.jelibaghu, michael.eley, alexander.palatnik, s200991, s201070}@th-ab.de*

[2] University of the Bundeswehr Munich, Werner-Heisenberg-Weg 39, 85579 Neubiberg, Germany; *oliver.rose@unibw.de*

SNE 36(1), 2026, 19-27, DOI: 10.11128/sne.36.tn.10764
Selected ASIM SST 2024 Postconf. Publication: 2025-06-30
Received Revised: 2026-01-15; Accepted: 2025-02-20
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna
ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. This paper explores the application of multi-agent reinforcement learning using the Proximal Policy Optimization (PPO) algorithm for resolving deadlocks in material flow systems with Automated Guided Vehicles (AGVs). A multi-agent strategy that optimizes the dynamics and interactions of multiple AGVs in real-time is implemented. The integration of the Population Based Training (PBT) algorithm from Ray enables continuous adaptation and improvement of learning processes. Subsequent modifications to the reward system have also been implemented to enhance the model's efficiency and effectiveness. The efficacy of the proposed approach is evaluated using a material flow simulation for a real industrial use case. The results demonstrate significant improvements in reducing collisions and increasing throughput within the system. This study highlights the potential of multi-agent reinforcement learning and specifically the PPO algorithm, to enhance the performance and efficiency of material flow systems with AGVs.

Introduction

Industries increasingly rely on automated systems for production and logistics, which brings about complexity in management, particularly with deadlocks where automated guided vehicles (AGVs) block each other, halting operations.

Traditional methods to resolve these deadlocks (Xu et al. 2014; Hussain et al. 2015), such as waiting or rerouting AGVs, are inadequate for larger systems due to scalability and adaptability issues. This paper proposes that Reinforcement Learning (RL), especially Multi-Agent Reinforcement Learning (MARL), can effectively address these challenges. It focuses on the Proximal Policy Optimization (PPO) algorithm, combined with Population Based Training (PBT) and targeted reward adjustments, to develop a robust and safe system for multi-agent environments.

The paper is organized as follows: Following a comprehensive literature review in Chapter 1, Chapter 2 describes the used simulation environment, the specific challenges faced throughout the project and how they were addressed. Finally, Chapter 3 presents and discusses the obtained results, while lastly Chapter 4 outlines perspectives for future research directions and the further development of the model.

1 Literature

1.1 Reinforcement Learning

Reinforcement learning (RL) is an area of machine learning in which an agent learns to make decisions by interacting with its environment. This learning paradigm has evolved considerably since the seminal work of Sutton and Barto (1998). At its core, RL is about developing a strategy or policy that maximizes the agent's cumulative long-term reward based on the actions it takes in different states.

The application of RL in complex decision-making tasks, has shown promise in the past. The ability of RL to learn from experience and adapt to dynamic environments makes it ideal for dealing with the uncertainties and complexity associated with such systems (Zhang et al. 2021; Choi et al. 2022).

The difference between Multi-Agent Reinforcement Learning (MARL) and Single-Agent Reinforcement Learning (SARL) lies in the number of agents that learn and make decisions in the environment.

While SARL focuses on scenarios in which a single agent controls all AGVs, MARL deals with situations in which multiple agents act simultaneously.

MARL is particularly relevant to the problem of deadlocks in logistics systems, as it considers the interaction of multiple AGVs and can develop strategies for cooperative behavior to effectively avoid or resolve conflicts and deadlocks [Stone & Veloso, 2000].

1.2 Proximal Policy Optimization (PPO)

The Proximal Policy Optimization (PPO) algorithm, presented by Schulman et al. (2017), is a further development in the family of policy gradient methods in RL. PPO aims to improve the stability and efficiency of the learning process by finding a balance between the agent's exploration ability and the utilization of what has already been learned.

In contrast to its predecessors, such as the Trust Region Policy Optimization (TRPO) algorithm, PPO offers a simplified yet effective method for policy optimization, making it particularly suitable for use in dynamic and complex environments such as those found in automated logistics systems.

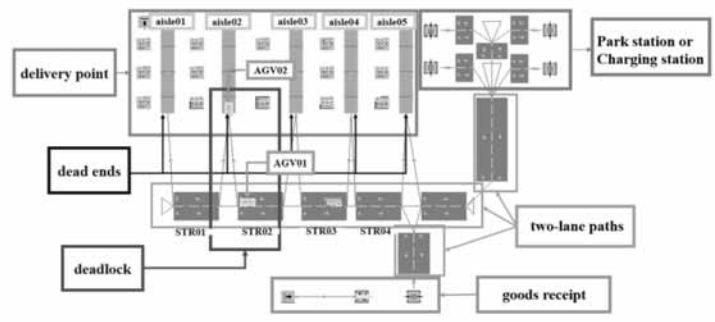

Figure 1: Illustration of a deadlock with three AGVs at the beginning of dead ends.

1.3 Population Based Training (PBT)

Population Based Training (PBT) is an approach for hyperparameter tuning that combines the advantages of genetic algorithms and hand-guided optimization. Developed by Jaderberg et al. (2017), PBT enables adaptive and time-efficient optimization of the learning processes of AI models. PBT dynamically adjusts hyperparameters as the model is trained, leading to a continuous improvement in model performance.

The main benefit of PBT lies in its ability to adapt to changing conditions within the training environment. Compared to conventional hyperparameter tuning methods, optimal configurations are achieved more effectively and quickly.

By combining these advanced techniques - MARL with PPO and dynamic adaptation through PBT - this project aims to develop a robust system for automated logistics that can avoid deadlocks while increasing the efficiency and reliability of the overall system.

2 Simulation Model

There has been a growing interest in using RL for deadlock resolution in intralogistics systems. In (Müller et al. 2022; Jelibaghu et al. 2023), the authors proposed a deadlock resolution method using a single RL agent. The agent was trained to learn how to resolve deadlocks by observing the states of the system and taking actions that lead to desired outcomes.

The agent was able to achieve high levels of deadlock resolution and collisions avoidance in (Müller et al. 2022).

As part of the research project, a real-world application for an AGV system was considered and modelled in Plant Simulation (cf. Figure 1).

The application is a high-bay warehouse with five aisles that the AGVs can only enter and exit from one side (dead ends). There are three AGVs available that have the task of moving pallets from the goods receipt, where the orders are created automatically and assigned to the AGVs (well known as dispatching), to the high rack. At the beginning the AGVs are located at the park station. A deadlock situation is shown in Figure 1. The deadlock occurs because the AGV01 currently located on the STR02 wants to enter aisle02.

At the same time, the AGV02 wants to leave aisle02. The simulation model is a digital twin of the logistics system, enabling scenario testing and performance optimization. It tracks material movement and component performance to identify bottlenecks and improvement areas.

The AGVs are controlled by an artificial neural network, which is implemented using Python code and connected to the simulation model and the AGV agents via the COM interface.

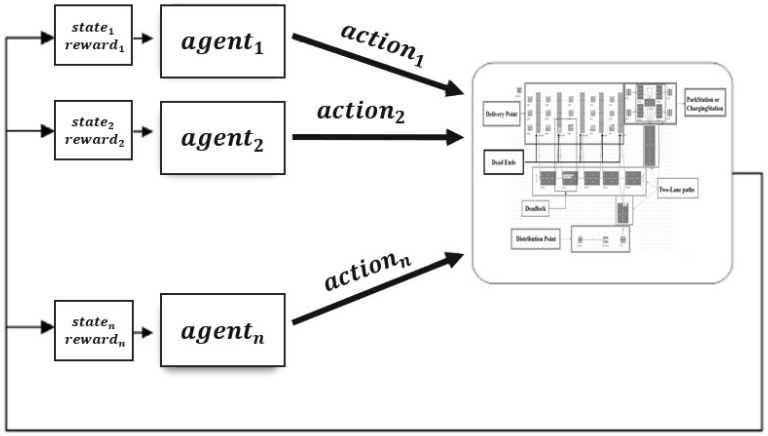

Figure 2: Agents in a multi-agent reinforcement learning (MARL).

Training configurations	
Episodes per training	300
Steps per epsiode	2000
Trials per training	5
Concurrent trials	1
Perturbation interval	5
Observation Space	
General	ID of the considered AGVs
For each agent *i*	*x* and *y* position of agent *i* Current speed of agent *i* *x* and *y* position of the target destination of agent *i*
Action Space	
Possible actions	MoveForward, MoveBackward, Stop
Reward-System	
Perform action	-1
Pick up order	+100
Complete order	+5000
Not-loaded agent exits aisle	+10
Agent enters occupied aisle	-10
Loaded agent exits aisle	-10

Table 1: Summary of initial action space and reward-system.

We decided to reduce the complexity of the environment, by reducing the number of agents to two. This adjustment aimed to provide a clearer view of how well our agent could learn and adapt, allowing for a more detailed observation.

To track the hyperparameters and the progression of rewards throughout the training, we utilized Weights & Biases (wandb).

This platform enabled us to gain insights into the performance of our model and observe the development of hyperparameters over the course of training. These adjustments resulted in clearer and more informative training progressions, allowing for more thorough analyses of the model's behavior and effectiveness.

In **Table 1** we present the initial setup of training configurations, action space and reward system, to show the baseline from which we began observing and adjusting the models performance and learning abilities.

As **Table 1** shows, at first the trials of the trainings were carried out serially and not in parallel. This had a considerable influence on the hyperparameters.

3 Experiments and Results

Our work builds on the work of (Müller et al. 2022; Jelibaghu et al. 2023). We propose a deadlock resolution method using a team of RL agents that is based on a real-world intralogistics system. We aim to evaluate our method on a number of different scenarios and demonstrate that it is able to achieve high levels of deadlock resolution and performance enhancement. In Figure 2 we show the MARL functionality.

Compared to Single Agent Reinforcement Learning (SARL), the MARL approach is a decentralized approach. This means that each AGV is considered as an independent decision maker.

In the initial phase of our study, we set out to intentionally provoke a deadlock scenario by initializing two or five agents and assigning the same drop-off location within the warehouse for the first ten orders. This approach quickly proved to be too aggressive or challenging, as it led to the AGVs consistently getting stuck. The high number of AGVs and task concentration offered little room for exploration. The resulting constant number of deadlocks highlighted the inadequacy of such a complex setup. Making it difficult to observe and understand the behavior of the reward system, the hyperparameters, and the learning efficacy of our agents.

The analysis is divided into five distinct sections to reflect the evolution of our model through various stages of adjustments.

Initially, the investigation begins with an examination of results obtained from the baseline setup, which represents our model's performance under initial conditions without any alterations to the reward system or hyperparameters tuning strategy. This foundational setup, shown in **Table 2**, is crucial for establishing a benchmark against which the impact of subsequent modifications can be measured.

After establishing a baseline understanding, the focus shifts to the outcomes following strategic adjustments to the reward system. This part dives into how modifying the reward parameters influences the learning trajectory and decision-making processes of the agents.

Hyperparameter	Search space function	Search space
Minibatch size	Randint	[4; 4000]
SGD-iterations	Randint	[3; 30]
Clip-parameter	Uniform	[0.1; 0.3]
Learning rate α	Uniform	[0.000005; 0.003]
KL-Coefficient	Uniform	[0.3; 1]
KL-target range	Uniform	[0.003; 0.03]
Discount factor γ	Uniform	[0.8; 0.9997]
GAE parameter λ	Uniform	[0.9; 1]
Value function coeff.	Uniform	[0.5; 1]
Entropy coefficient	Uniform	[0; 0.01]

Table 2: Hyperparameter search spaces and functions.

The following part of this chapter explores the effects of refining the tuning mechanism. It examines how the implementation of a more dynamic and responsive tuning approach impacts the model's performance.

Lastly, we show the ability and the potential of our model to reduce the number of collisions while increasing the number of completed orders. This gives an outlook on the capability of the model to successfully resolve deadlocks.

2.1 Baseline Setup

We begin by examining the performance of one agent (SARL) before expanding our analysis to a scenario involving two agents (MARL). To be clear, we built the following scenario. First we ran and analyzed 1 agent in the environment and then we took the same model and increased it to 2 agents. **Figure 3** shows the result of the analysis. We use the result of the single agent as a benchmark. We analyze the maximum rewards achieved during the training. This metric is indicative of the agent's ability to optimize its behavior within the environment, with the highest reward in a series of trials representing the peak efficiency reached by the agent.

Out of five trials conducted during this phase, only the best-performing trial is considered for detailed analysis. This selection criteria ensures that we capture the agent's optimal performance potential under the baseline settings (cf. **Table 1**).

In the initial graphical analysis, contrasting performance outcomes between single-agent and dual-agent setups provide insightful revelations about their operational efficiency within the simulation. The **Figure 3** refers to 1 agent and 2 agents results. 2 agent refers precisely to the accumulated reward of 2 agents. Initially, it might be assumed that the single agent would outperform due to its sole occupancy of the environment, facing no competition or interference.

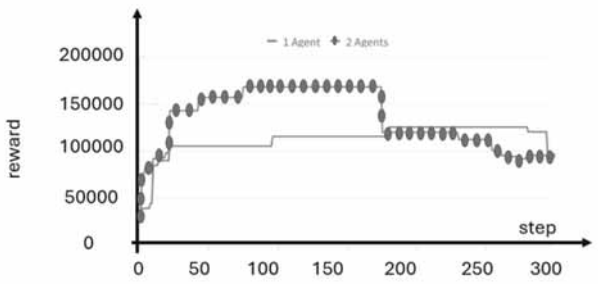

Figure 3: Max reward during training with multi agents.

However, our findings show a different picture; the dual-agent setup achieved a higher maximum reward (cf. **Figure 3**). It demonstrates the cumulative advantage of having double the number of steps available compared to a single agent. This increased action potential, combined with effective learning behavior, indeed resulted in higher rewards for the dual-agent training scenario. It emphasized the significance of collaborative efforts in navigating complex environments.

Despite having twice as many steps at their disposal, the fact that the dual agents did not achieve double the reward of a single agent can be directly attributed to the escalated complexity introduced by their interaction. This complexity presents both challenges and opportunities for learning and optimization, highlighting the delicate balance between cooperation and competition in a shared environment.

Upon closer examination across all five trials for each training setup, it becomes evident that the two-agent model not only outperforms the single-agent framework in terms of maximum rewards but also in cumulative mean reward.

This shows that the enhanced learning outcomes facilitated by multi-agent interactions are significant. The shared policy learning dynamic allows the system to benefit from a richer set of information for policy adjustments. This increases the likelihood of achieving successful, stable training outcomes.

2.2 Reward System

Table 3 shows the updated reward system, detailing the strategic modifications made to enhance navigation in the simulated environment. These changes were strategically implemented to swiftly move agents out of the aisles, thereby reducing potential deadlocks and easing traffic congestion.

This refinement entailed modifying the existing movement incentives for "*MoveForward*", "*MoveBackward*", and "*Stop*" actions, which are integral to the agents' navigation through the simulation environment.

The standard penalty assigned to any step taken remained set at -1, preserving the agents' motivation to minimize unnecessary movements. However, to promote more efficient behavior post-order completion, executing a "*MoveForward*" or a "*Stop*" action was assigned a larger penalty of -5, while a reward of +2 was allocated for a "*MoveBackward*" action.

Reward-System	
Perform action	-1
Unloaded AGV performs *MoveForward* on aisle	-5
Unloaded AGV performs *MoveForward* on aisl	-5
Unloaded AGV performs *MoveBackward* on aisle	+2
Pick up order	+100
Complete order	+5000
Unloaded AGV exits aisle	+25
Loaded AGV exits aisle	-10
AGV enters occupied aisle	-10
Unloaded AGV enters aisle	-50

Table 3: Adjusted reward-system.

This careful calibration of rewards and penalties was designed to maintain a balance where agents are discouraged from counterproductive movements without overshadowing the adverse implications of such actions, thereby sustaining the learning impact.

To further encourage the desired behavior, the action of successfully exiting an aisle was assigned a higher reward of +25. Conversely, to deter agents from exploiting this system by repeatedly entering and exiting the aisle without carrying a load, a hefty penalty of -50 was imposed on unloaded travel on the street. This modified incentive system was implemented to guide the agents towards behaviors that enhance overall efficiency in the logistics environment.

Figure 4 illustrates an improvement in the maximum reward output compared to **Figure 3**, which represents the results from the initial setup. This enhancement confirms that the modifications to the reward system have been effective, showing a positive impact on the overall performance.

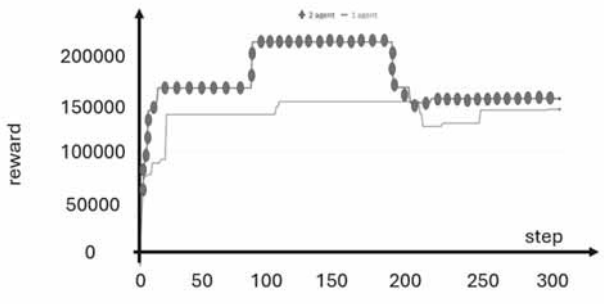

Figure 4: Max reward of training with new reward setup.

2.3 Hyperparameters

In our exploration of hyperparameters and the implementation of the tuning process, the utilization of the Population Based Training (PBT) algorithm played a pivotal role. Despite careful inclusion of all relevant parameters within the PBT tuner, along with the definition of search spaces and corresponding search functions, it was noted that several values remained at their initial setting throughout the training period. This phenomenon of no adaption applied to all hyperparameters except for the clip coefficient. Unexpectedly, the clip coefficient exhibited variations that extended beyond the predefined search space boundaries. At the start of each trial, a random value for lambda was selected from within the search space, establishing the initial conditions under which each trial would proceed.

To leverage the full capabilities of the PBT mechanisms, we adjusted the number of concurrent trials to five, allowing for simultaneous training of all trials. This setup positioned each trial as an integral component of the PBT's population, fostering a dynamic learning environment. The perturbation interval, set at every five episodes, served as a benchmark for evaluating and comparing trials.

This interval determined the feasibility of continuing a trial based on its performance. This necessitates adjustments to align with more successful trials or terminating trials that failed to show promise. It should be mentioned here that training the trials in parallel leads to significantly longer training times and to a partially incomplete recording of the courses in wandb.

This strategic approach aimed to optimize the learning process by facilitating real-time adaptations and fostering a competitive yet collaborative environment among the trials. The performance outcomes, captured in Figure 5, illustrate the impact of these adjustments on the model's learning efficiency.

Due to gaps in data collection for these parallel trials, the data of some trials was not fully tracked by wandb. Follow several reasons can be attributed: Technical Issues, System Overload, Network Interruptions, Human Error and Software Bugs. Nevertheless, the results in Figure 5 showcase clear improvement in the collected max-reward, compared to the baseline setup with serial trials (**Figure 3**). These results suggest that parallel training, despite requiring longer training durations, has a positive impact on the agents' performance. Such high-reward instances demonstrate the potential benefits of conducting training in parallel. It demonstrates that this approach can significantly enhance the learning outcomes and operational efficiency.

The primary advantage of conducting parallel trials in PBT lies in the dynamic adjustment of hyperparameters in real time. While most parameters remained constant during serial trials, the training involving parallel trials saw dynamic adaptations. This leads and enhances learning behaviour, making the training process more effective. By terminating less promising trials, the algorithm, ensures that no resources are wasted on prolonging the training time unnecessarily. Thereby, it puts the focus on the most promising configurations.

This approach allows for the rapid identification of superior hyperparameter configurations compared to manual tuning, The capability to dynamically adjusting and refining the training process in real time represents a pivotal shift towards more agile and responsive agent training methodologies.

Figure 6 provides a detailed visualization of the hyperparameter values across all trials. It is evident that many trials exhibit similar or identical hyperparameter values, which is consistent with the observation from serial trainings that hyperparameters tend to remain constant. Nevertheless, the trials that underwent dynamic adjustments generally show improved performance, reinforcing the advantage of real-time hyperparameter tuning.

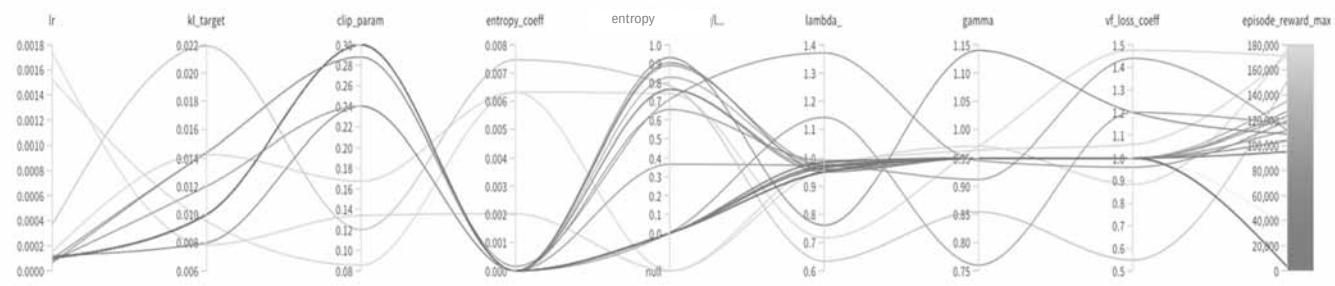

Figure 6: Hyperparameter ranges and their impact on the max reward.

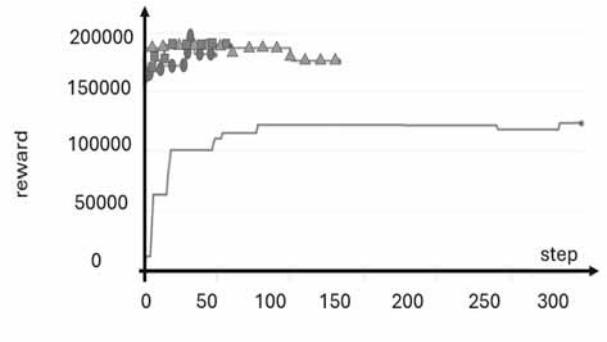

Figure 5: Max reward of training with 2 agents and concurrent trials.

However, this does not negate the possibility of achieving good performance in serial trials, as previously demonstrated.

When analysing the hyperparameters of the parallel training trials, it is essential to recognize that the values presented represent only the average due to the dynamic adjustments made during the process. From these averages, it is observable that a higher entropy coefficient, which introduces more randomness into the decision-making process, can be beneficial to the results. An increase in entropy generally encourages exploration, which can prevent the agents from getting stuck in suboptimal behaviors.

Furthermore, a slight uptick in the learning rate, paired with a reduced clip parameter, appears to be advantageous. The learning rate determines how quickly the model adapts to new information, while the clip parameter helps in stabilizing the policy update.

Adjusting these parameters could lead to a more efficient learning process by balancing the rate of adaptation with the stability of learning. However, pinpointing the exact values that yield the best performance is challenging, as a wide array of configurations have resulted in favorable outcomes. Therefore, while certain trends in hyperparameter adjustments can be identified as generally positive, the diversity in effective configurations emphasizes the complexity and adaptive nature of the learning environment. This variability shows the need for a comprehensive hyperparameter search when tailoring the model to maximize performance.

2.4 Collision avoidance

In the final section of the analysis, we demonstrate the capability of our model to not only avoid collisions but also to reduce their occurrence over the course of training, while simultaneously enhancing the throughput of completed orders.

We present three Figures to illustrate these developments: Figure 7 displays the reduction in the number of collisions throughout the training sessions.

Figure 8 shows the corresponding increase in the number of successfully completed orders. These completed orders lead to the reward shown in Figure 9.

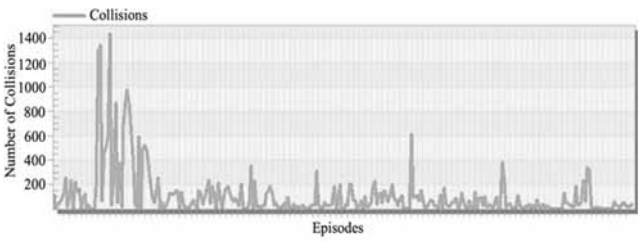

Figure 7: Number of cumulative collisions of 2 agents during training process.

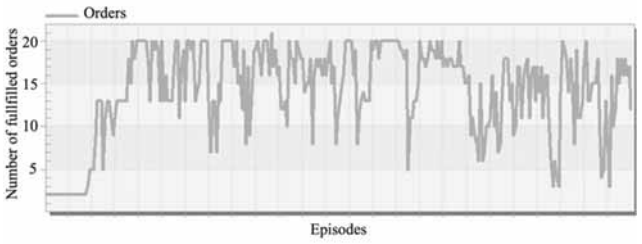

Figure 8: Number of cumulative completed orders of 2 agents during training process.

These figures clarify that the MARL approach effectively enhances operational efficiency by increasing throughput and minimizing disruptions caused by collisions.

Furthermore, these results indicate that the model possesses significant potential to avoid and resolve deadlocks, as collisions and deadlocks present very similar scenarios within our simulations.

This demonstrates substantial improvements in both safety and efficiency of the AGV-fleet.

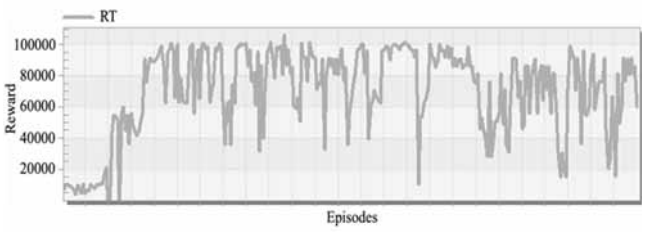

Figure 9: Accumulated reward of 2 agents during training process.

2.5 Testing phase

Eventually, we wanted to verify the policy our agents have learned over the period of training. Therefore, we established checkpoints to capture and store the policy and hyperparameter configurations learned by the agents at specific training stages. By reloading these checkpoints into a model, we were able to verify that the agents were indeed learning.

During the 100-episode learning phase the agents were still allowed to continue exploring, with the data loaded from the checkpoints serving as starting points for further learning and adaptation.

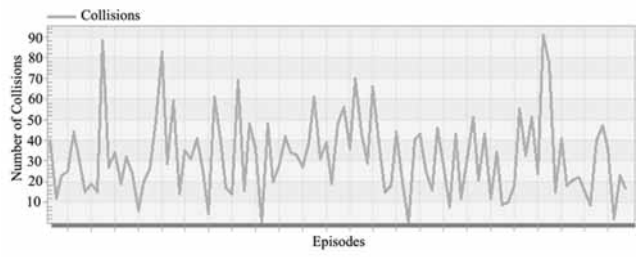

Figure 10: Number of cumulative collisions of 2 agents during testing phase.

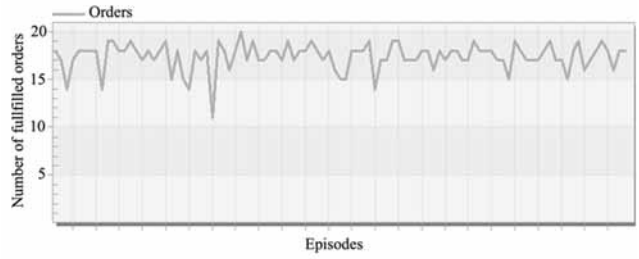

Figure 11: Number of cumulative completed orders of 2 agents during testing phase.

Figures 10 to 12 offer a clear visualization of the outcomes from the checkpoints applied during our testing phase. Figure 10 illustrates the cumulative number of collisions, which have remained consistently low from the beginning of the testing phase, suggesting that the collisions are well-managed throughout the training.

Figure 11 shows the cumulative number of completed orders, where it is evident that the orders are completed at a high rate, consistently ranking in the upper performance tiers of our training data.

Lastly, Figure 12 shows the cumulative rewards occrued, which are also high from the outset of the testing phase.

These visuals collectively demonstrate that the checkpoints contain effective policy and hyperparameter configurations, and that the agents did not require a reset or additional exploration phases to achieve these results. This underscores the robustness and efficiency of the learning mechanisms we have implemented.

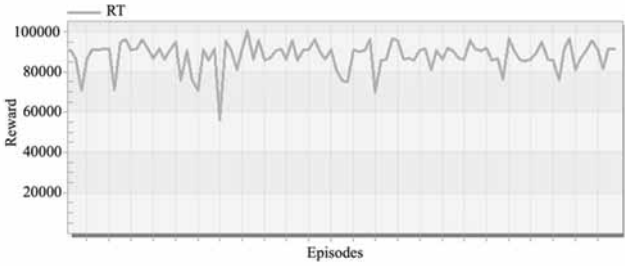

Figure 12: Sum of collected reward of 2 agents during testing phase.

3 Conclusion and Outlook

In summary, this study has demonstrated that the Proximal Policy Optimization (PPO) approach coupled with Population Based Training (PBT) for hyperparameter tuning is suitable for efficiently coordinating AGV fleets to reduce collisions and deadlocks and thus throughput to increase.

Possible extensions of the approach address the question of how to increase the number of agents. This will allow us to better understand the scalability of our approach and to refine the interplay between agents. Furthermore, we plan to experiment with different configurations of the perturbation interval, a critical mechanism in PBT that can significantly influence learning outcomes.

In parallel, further adjustments to the reward system are on the horizon. These modifications aim to enhance the learning process even more, and we intend to evaluate the impact of these changes, especially when combined with parallel trials. With these adjustments, we anticipate uncovering even richer insights into the model's performance.

While our findings have shown promise, the capabilities of our model and the process of refining its performance is ongoing. There are still many opportunities to further improve the model and gain deeper understanding, driving the evolution of AGV coordination in logistics.

Publication Remark

This contribution is the revised version of the conference version for

ASIM SST 2024 - 27th Symposium Simulation Technique - Munich 2024,

published in

ASIM 2024 Tagungsband Langbeiträge
ARGESIM Report 47, e-ISBN 978-3-903347-65-8,
Volume DOI 10.11128/arep.47
Article DOI: 10.11128/arep.47.a4705, p 203-210.

References

[1] Xu J, Zheng Z, Lyu M. CGA-based deadlock solving strategies towards vehicle sensing systems. EURASIP Journal on Wireless Communications and Networking 2014 (2014). DOI 10.1186/1687-1499-2014-214

[2] Hussain S, Kumar S, Janardh G. Deadlock avoidance and re-routing of automated guided vehicles in flexible manufacturing systems. International Journal of Advances in Production and Mechanical Engineering (IJAPME) 1 (2015) 4.

[3] Sutton RS, Bart, AG. Reinforcement Learning: An Introduction. MIT Press (1998).

[4] Zhou T, Tang D, Zhu H, Zhang Z. Multi-agent reinforcement learning for online scheduling in smart factories, Robotics and Computer-Integrated Manufacturing, Volume 72, 2021, https://www.sciencedirect.com/science/article/pii/S0736584521000855

[5] Choi H-B, et al. MARL-based Optimal Route Control in Multi-AGV Warehouses. 2022 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Jeju Island, Korea, Republic of, 2022, pp. 333-338. DOI 10.1109/ICAIIC54071.2022.9722643

[6] Schulman J, Levine S, Abbeel P, Jordan M, Moritz P. (2017). Proximal Policy Optimization Algorithms. arXiv:1707.06347.

[7] Jaderberg M, Dalibard V, Osindero S, Czarnecki WM, Donahue J, Razavi A, Vinyals O, Green T, Dunning I, Simonyan K, Fernando C, Kavukcuoglu K. (2017). Population Based Training of Neural Networks. arXiv:1711.09846.

[8] Stone P, Veloso M. Multiagent systems: A survey from a machine learning perspective. (2000). Autonomous Robots, 8(3), 345-383.

[9] Müller M, Reggelin T, Kutsenko I, Zadek H, Reyes-Ruiano. Towards deadlock handling with machine learning in a simulation based learning environment. Proceedings of the 2022 Winter Simulation Conference 11-14 December 2022 Singapore, 2022, pp. 1485-1496.

[10] Jelibaghu M, Eley M, Palatnik A. 2023. Simulation-Based Resolution of Deadlocks in Automated Guided Vehicles using Deep Reinforcement Learning. ASIM Fachtagungen Simulation in Production and Logistics, September 2023 Technische Universität Ilmenau, Germany.