Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

SIMULATION STUDY ANALYZING THE GENERALIZATION OF A MULTI-AGENT REINFORCEMENT LEARNING APPROACH TO CONTROL GANTRY ROBOTS

Horst Zisgen[1], and Jannik F. Hinrichs[1]

[1]Dept. of Math. and Nat. Sci., Darmstadt University of Applied Sciences, Darmstadt, GERMANY

ABSTRACT

Industry 4.0 forces a significant transition in the field of manufacturing and production planning and control. A corner stone of Industry 4.0 scenarios is the ability of control systems to adapt autonomously to changes on the shop floor. Reinforcement Learning is considered as an approach to achieve this target. Consequently, the control strategies of the agents trained by Reinforcement Learning need to generalize in a manner that the agents are able to control modified production systems they are not directly trained for. This paper presents an evaluation of the generalization properties of a decentralized multi-agent Reinforcement Learning algorithm for controlling flow shops using complex gantry robot systems for automated material handling. It is shown that the corresponding agents are able to cope with variations of the production and gantry robot system, as long as these variations are within realistic boundaries, and thus are suitable for Industry 4.0 scenarios.

1 INTRODUCTION

Industry 4.0 forces a considerable transition in the field of manufacturing. On the one hand, this opens up potentials for improving efficiency, but on the other hand, it requires a substantial adaptation of processes in planning and operation. A key requirement of Industry 4.0 is that production control algorithms should be able to adapt autonomously to changing conditions on the shop floor. This is e.g. reflected in acatech's Industrie 4.0 Maturity Index (**?**). As a consequence, traditional planning processes need to be accelerated and optimized and planning systems need to be adjusted (**?**). In those traditional environments the industrial engineering team had a reasonable lead time to adjust scheduling or dispatching rules to the changed settings of machines or material handling processes. In an ideal Industry 4.0 scenario, this should happen automatically, without any time delay and, of course, without any loss of manufacturing efficiency. Artificial Intelligence (AI) and Machine Learning (ML) are changing the manufacturing landscape fundamentally (**?**) and are considered as an approach to meet these requirements. In particular, Reinforcement Learning (RL) combined with simulation is becoming more attractive in the area of production planning and control (**?**). As consequence, there is a large body of literature on the application of RL to production scheduling and control. Good surveys of the most recent work can be found e.g. in **?** and **?**.

In RL, an agent is trained to control a system by interacting with its environment. During training, the agent learns a policy that enables it to make autonomous decisions based on its observations. For further details about RL refer to **?**. The advantage of RL agents is that they can usually handle stochastic interference better than heuristics or fixed dispatching rules.

The purpose of this paper is to study such an Industry 4.0 scenario based on a flow shop utilizing gantry robots as an automated material handling system controlled by RL agents. Although there is a vast body of literature regarding on the use of RL in production control, it mainly concerns the job shop scheduling. In contrast, only a few papers present RL approaches for flow shop scheduling, particularly in combination with controlling the corresponding material handling system, such as a gantry robot system. The literature review by **?** references only a few papers concerning RL approaches for flow shops, one of which includes robot control. For example, **?** and **?** describe tabular Q-learning approaches. However, these only consider one robot and are not efficient enough to control complex, real-world gantry systems because tabular Q-learning does not scale. In **?**, a DQN-based single-agent approach is presented that can efficiently control a single gantry.

To analyze the Industry 4.0 scenario, we first briefly present a decentralized multi-agent Reinforcement Learning (MARL) algorithm for controlling complex gantry robot systems with multiple gantries. As shown in **?**, the MARL approach is much faster than the comparable single-agent approach given in **?**. Secondly, we analyze whether the agents satisfy the Industry 4.0 requirements in terms of flexibility and autonomous adaptation to changes in the production environment, since it is insufficient for RL agents to efficiently control systems only under the exact conditions on which they were trained. Rather, it is crucial that their control strategies are flexible enough to generalize (**?**). In the gantry robot context, generalization can be viewed from two different perspectives.

1. **Process generalization** The agents are trained with a specific set of parameters related to work stations, gantry robot speed, processing times, and so on. Generalization can be interpreted as the ability of the agents to control the gantry robot system even in cases where the "real parameters" are different from those the agents were trained with. This means that the agents are able to operate in a more general environment that is broader than the specific training environment. As a result, the RL agents are able to respond autonomously to changes on the shop floor without having to be retrained.
2. **Training generalization** The agents themselves are characterized by several hyperparameters, which influence both their behavior and performance. From this perspective, a relevant question arises: do the selected hyperparameters generalize sufficiently well? In particular, if the agents need to be retrained, due to significant changes in the gantry robots system - such as the number of work stations or the number of gantry robots - can the same set of hyperparameters still ensure a training that is equally efficient and robust as with the initial configuration?

In this paper, we analyze the characteristics of MARL agents with regard to the their process generalization ability. The analysis is based on intensive simulation studies partly presented in this paper. Accordingly, the paper is organized as follows. In Section 2 an overview of the MARL algorithm for gantry robot systems is provided. The experimental setup for the performed analysis of the generalization properties is presented in Section 3. The evaluation results are delineated in Section 4. Finally, in Section 5 a critical summary and an outlook for future research needs are given.

2 MARL FOR GANTRY ROBOTS

This section is composed of two parts. The first part is an overview of the structure of the used gantry robot systems. The second part depicts the utilized MARL approach for training the agents.

2.1 Gantry Robot System

Gantry robot systems represent a conventional method for automating material handling in high-volume manufacturing lines organized as flow shops. In this context, the machines or work stations are arranged in a series according to the process flow. These work stations are enumerated in alphabetical order $A, B, C, \dots$, whereby A is the station next to the input conveyor. A work centers compromises of one or more work stations performing the same operation. For example, the production cell illustrated in Figure 1 features three work centers composed of the stations $\{A,B\}, \{C,D\}$ and $\{E,F\}$.

Above the machines and both conveyors a rail is mounted. The gantry robots move along this rail to handle the transport of the workpieces downstream through the production cell according to the designated process flow. The gantry robots $g_i, i = 1,2,\dots,m$ are enumerated in numerical order, whereby the gantry next to the input conveyor is g_1 and the gantry next to the output conveyor is g_m. Typically, there are two distinct types of gantries: I-style and H-style gantries. An I-style gantry is characterized by its singular

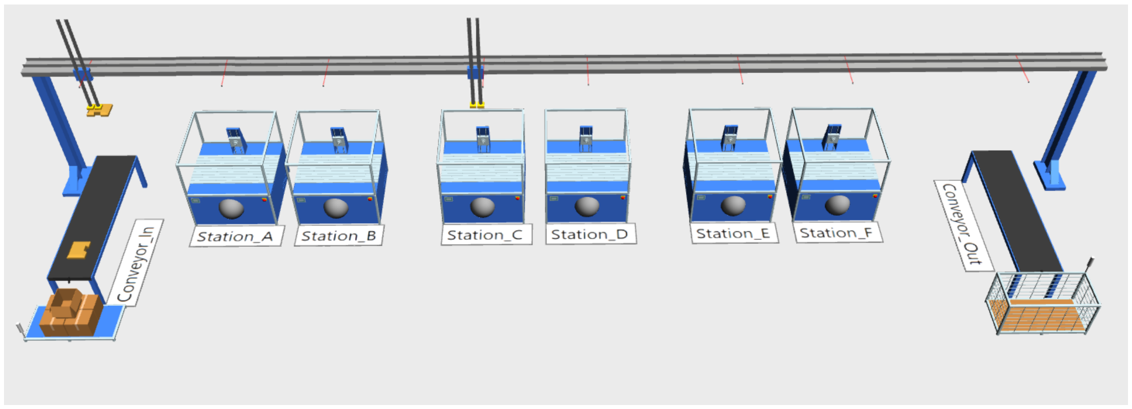

Figure 1: Simulation model of a production cell with two H-style gantries, one input conveyor, three work centers comprising of two work stations each and one output conveyor. The model is visualized in PlantSimulation.

gripper, which only allows the transport of one work piece at the time. In contrast, H-style gantries have two grippers, allowing the gantry to transport two work pieces simultaneously and performing a fast work piece swap at a work station. A fast swap is the situation, when a gantry picks up a finished workpiece from a work station and then directly unloads an unprocessed workpiece form its other gripper into the now empty work station.

The production line under consideration does not include buffers at the work stations. New unprocessed workpieces are supplied to the input conveyor. Upon arrival, the parts are queued on the input conveyor and the gantry robot g_1 can pick them up. Analogously, finished workpieces have to be moved and placed on the output conveyor by the last gantry robot g_m. The output conveyor transports them to a stock or another production stage.

2.2 MARL Algorithm

When complex production cells with multiple gantry robots are modeled as an RL problem, the state-action space and the necessary training duration increases rapidly. In order to address this challenge, the decentralized MARL approach introduced in **?** is employed. The MARL approach assigns for each gantry robot its individual RL agent, which are all trained separately. In consideration of the underlying structure of the associated flow shop, the implementation of an overarching mechanism to synchronize the individual agents is not necessary. The maximization of the throughput by each agent within the section controlled by the agent is ensuring the maximization of the throughput of the entire production line. This methodology enables the a very efficient training of multiple agents, with each agent representing a distinct gantry robot.

The concept of operational areas is introduced in **?** in order to reinforce the aforementioned concept. This concept is motivated by the physical constraints of the gantry's cruising range, which are a result of the drag chain that connects the robot to the power supply. The operational areas constitute a subset of the work stations that a gantry robot is capable of approaching. It is necessary to define these areas in advance, as they influence the internal architecture of the RL agents. Consequently, the state space of each agent is required to contain solely the status of each work station within its operational area. This work station status is comprised of the three features: whether the machine is occupied, the processing has finished, and whether a failure has occurred. With regard to the controlled gantry robot itself, the state space includes the position of the gantry and the state of each gripper, thereby delineating the subsequent process step of the loaded workpiece. As last part of the RL agent state space the current position and executed action of the neighboring gantries is included. The neighboring gantries of gantry g_i are defined as $\{g_{i+1}\}$ for $i = 1$, $\{g_{i-1}, g_{i+1}\}$ for $i = 2, \ldots, m-1$, and $\{g_{i-1}\}$ for $i = m$.

At each decision point t the simulation requests a new action for only one RL agent, as gantry robots act in a asynchronous manner. The corresponding RL agent then selects one of the following actions depending on its current policy:

- move the robot to a designated target work station or conveyor
- picking up a workpiece from either the input conveyor or a work station
- dropping a currently loaded workpiece into a work station or at the output conveyor
- waiting until a system change triggers a subsequent decision point

In order to prevent needless actions or even those that would result in collisions of gantries, it is feasible to restrict the action space or to penalize those actions through a appropriate designed reward function. The corresponding reward function is given by

$$R_{t+1}^{(i)} = \begin{cases} 5 & \text{finished workpiece delivered at the output conveyor} \\ 2 & \text{workpiece unloaded correctly into a work station} \\ 1 & \text{workpiece picked up by a gripper correctly} \\ -1 & \text{unneeded stopover} \\ -3 & \text{waiting while blocking another gantry robot} \\ -5 & \text{collision-causing action} \\ 0 & \text{other} \end{cases} .$$

To guarantee the appropriate credit assignment, the rewards get only assigned to the agent i associated with the gantry the action $A_t^{(i)}$ is requested for.

The agents choose their actions according to a policy π. The action-value function $q_\pi(a,s)$ for a given policy π is defined as the expected sum of discounted future rewards the agent gains when starting at time t in state s (i.e. $S_t = s$), choosing action a at time t (i.e. $A_t = a$) and following from there policy π, i.e. $q_\pi = E(\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s, A_t = a)$, $\gamma \in (0,1)$. The goal is to determine the optimal policy π^* which maximizes $q_\pi(s,a)$ for every state-action pair (s,a). However, this becomes analytically intractable for large state-action spaces, so one must approximate q_{π^*}. Q-learning (**?**) is an approximation approach that iteratively approximates q_{π^*} with a step-by-step simulation and the update procedure

$$Q(S_t,A_t) \leftarrow Q(S_t,A_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(S_t,a) - Q(S_t,A_t) \right] ,$$

with the learn rate $\alpha \in (0,1]$. The MARL algorithm, that is used to control the gantry robots, employs basically the Q-learning update procedure, however the action-value function $q_\pi(a,s)$ is approximated by a neural network. This approach is called Deep Q-Network (DQN) (**?**). Each RL agent in the MARL algorithm is set up as a DQN agent, that is trained individually using the update formula as follows

$$Q(S_t,A_t;\theta) \leftarrow Q(S_t,A_t;\theta) + \alpha \big[R_{t+1} + \gamma^{1+d_{t+1}} \max_a Q(S_{t+1},a;\theta^-) - Q(S_t,A_t;\theta) \big].$$

Thereby refers θ to the prediction network, i.e. the neural network used as function approximation of the Q-function responsible for choosing the actions during training (also called behavior network). θ^- denotes the target network, i.e. the neural network used as approximation of the target value to determine the error of the updating step. In contrast to the plain discount factor γ the update formula employs $\gamma^{1+d_{t+1}}$, in which d_{t+1} represents the duration of the selected action A_t. The exponent $1 + d_{t+1} \geq 1$ makes shorter actions more preferable. Eventually, at each decision point t, the related DQN agent chooses the action A_t that provides the highest action-value $Q(A_t, S_t; \theta)$ for the current state S_t.

3 EXPERIMENTAL SETUP

This section presents the experimental setup for analyzing the generalization properties of the RL agents described in Section 2. The starting point of the analysis is a base model. The RL agents that are trained to control that specific base model, are referred to as *Base Model Agents*. Subsequently, the settings of the production system or the gantry robots are varied, and the RL agents are validated concerning their performance with these modified models. Ideally, the RL agents are capable of controlling the gantries in the modified systems in a stable manner. In other words, it is expected that they are able to make reasonable but not necessarily optimal decisions in each situation.

3.1 Base Model

For the presented analysis two different base models are used. The first base model I is illustrated in Figure 1 and depicts a production cell with two H-style gantries and three different process steps p_1, p_2, p_3 performed by a work center each, that each compromises two identical work stations. In other words, process step p_1 is performed at work center A or B, p_2 at C or D and p_3 at E or F. The second base model II is a more complex production cell with three H-style gantries and five different process steps $p_1, \dots, p_5$ performed at the five work centers $\{A,B\}, \{C,D\}, \dots, \{I,J\}$. In both models the supply of new workpieces at the input conveyor is stochastically distributed. The processing time at each work station is deterministic and the machine break downs take place at random and for a stochastically distributed period of time (Table 1). The gantry robots have a cruising speed of $2\,m/s$, whereby acceleration and deceleration are neglected. The distances between the work stations are presented in Table 2.

Table 1: Parameters describing the dynamic behavior of the simulation model for the reference models.

Parameter	Value
Interarrival Interval at Input Conveyor	$U[5,15]\,s$
Station Processing Duration	$20\,s$
MTBF	$270\,s$
MTBF Distribution	$exp(\text{MTBF}^{-1})$
MTTR	$30\,s$
MTTR Distribution	$\text{Erlang}((2*\text{MTTR})^{-1}, 2)$

Table 2: Station position along the rail in meters for the base models and their variation.

Model	In	A	B	C	D	E	F	G	H	I	J	Out
I: Base	0	3	5.5	9	11.5	15	17.5	-	-	-	-	21.5
I: Variation	0	5	13	17	20	22.5	28	-	-	-	-	31
II: Base	0	3	5.5	9	11.5	15	17.5	21	23.5	27	29.5	33
II: Variation	0	7.5	15	22	25	30.5	34	40.5	43.5	46.5	49	55

For the base model I the agents were trained over $N_\tau = 2\,500$ distinct seeded episodes simulating the entire production system for duration of $T_\tau = 1800s$ each. For the base model II the training duration was increased to $N_\tau = 4\,000$ and $T_\tau = 3600s$ to address the more complex setup. All validations for both models are performed with $N_v = 100$ runs simulating the production system for $T_v = 7200s$ each. The simulated time is increased for the validation runs, on one hand, to confront the agents with new situations that may not have occurred during the shorter training simulations. On the other hand, each run initiates with an empty production cell, resulting in a warm-up phase with reduced throughput. The duration of these warm-up phases varies depending on the individual model settings. To reduce the impact of the warm-up phase, the two-hour duration was chosen as a compromise for our validation setup. The remaining training hyperparameters are set as described in **?**.

3.2 Model Variations

In order to identify the limitations of the generalization capabilities of the agent, mostly extreme variations are made to the base models, even if these are very unlikely in practice. The performed model variations are grouped into the two categories: *gantry variations* and *process variations*.

Gantry variations

- Reducing the gantry cruising speed by $25\,\%$ to $1.5\,m/s$
- Reducing the gantry cruising speed by $50\,\%$ to $1.0\,m/s$
- Modified distances between the work stations (Table 2)

Process variations

- Doubling the process times to $const(40)$ seconds
- Mixing process times per work center, i.e. the work center take on average $20\,s$ per work piece but the process times of the work stations in the work center are not the same (e.g. $10s$ at A and $30s$ at B)
- Process times vary $\pm 25\%$ around the mean according to a symmetric Triangle $(15, 20, 25)$ distribution
- Process times vary $-25\%/+100\%$ around the mean according to a right skewed Triangle $(15, 20, 40)$ distribution

4 RESULTS

In the following the results based on the validation runs are presented and discussed. Figure 2 presents an overview of the validation runs performed for the first base model I and the *gantry variations*. The validation run number is shown on the x-axis, while the y-axis shows the throughput per hour achieved in the corresponding validation run. It is evident that the throughput varies from the base model's throughput in each scenario, due to the altered settings of the gantry robots. Nevertheless, the validation of the three variations show that the RL agents are capable of controlling the modified systems in a stable manner.

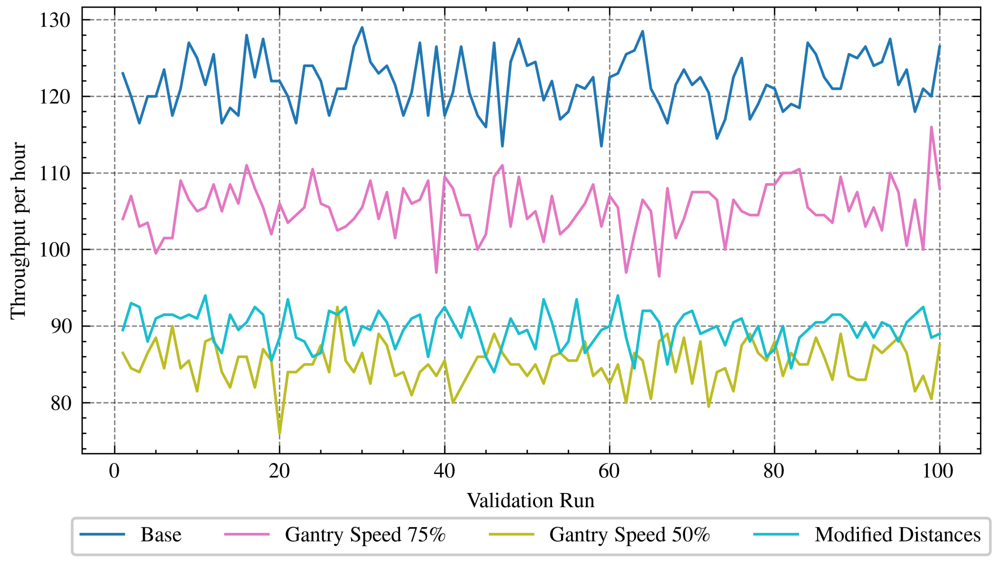

Figure 2: Results of validation runs for the base model I (2 H-style gantries; 3 process steps) with *gantry variations*.

In order to verify that the controls are not just stable but also perform well with regard to the throughput, the agents were also trained directly on the modified model, i.e. with the changed gantry speeds and distances. As expected, these agents yield a higher throughput. The corresponding throughput values are shown in Table 3. But it can be seen, that when the variations are not too extreme, the throughput achieved in the variation is close to the throughput achieved in the case the agent is trained directly with the modified parameters. When reducing the gantry speed down to 75% the throughput per hour of the variation is in average 105.4. This is approximately 95% of the average throughput per hour achieved by the directly trained agents. This is fairly reasonable, since the reduction by 25% is not a small matter. Even in the edge case of reducing the gantry speed by half, the agents achieve about 89% of the average throughput per hour that the directly trained agents are able to reach. This is still remarkable in view of the extreme variation of the base model. Further, it has to be mentioned, that the direct training of the variation with 50% reduced gantry speed results in an even lower throughput and more unstable policy, when the same hyperparameters are used. In order to achieve a stable policy with higher throughput of an average of 96.1 parts per hour, it turned out that it is necessary to extend the number of training episodes up to $N_\tau = 4000$. This is caused by the reduced number of events (i.e. actions carried out) during the simulated training period due to the slower gantry speed. This must be compensated for by a higher number of training episodes in order to provide the agents a sufficient learning experience. Also in case of an extremely changed topology of the production line, in terms of the distances between work stations, the agents still achieve remarkable 86% of the mean throughput per hour the directly trained agents delivers.

Table 3: Mean throughput per hour with CI-95% for the validation runs of the *gantry variations* of the base model *I* and the directly trained modified model.

Variation	Base Model Agents	Directly Trained Agents	Extended Trained Agents
Gantry Speed 75%	105.4 [104.75, 106.06]	110.4 [109.78, 111.05]	-
Gantry Speed 50%	85.1 [84.58, 85.62]	65.8 [57.43, 74.18]	96.1 [94.69, 97.51]
Modified Distances	89.7 [89.24, 90.15]	103.7 [103.09, 104.37]	-

A similar behavior is observed for the base model *II* (Figure 3). Only in the edge case of a reduction of the gantry speed by 50% the RL agents are in about a third of the validation runs not able to achieve a reasonable throughput. This is mainly due to deadlocks, in which the agents repeatedly select wrong actions, thereby preventing any further progress. One potential root cause of this phenomenon is that the extreme reduction in the gantry speed, and therefore prolonged travel durations. This leads to state combinations that were infrequent or nonexistent during the base training with the standard gantry speed. Thus, the DQN could not be accurately trained for these cases. Training the three agents directly with the modified model yield a stable policy as well.

The validation results for the *process variations* on base model *I* (Figure 4) show that the RL agents can adapt to those variations in a reasonable manner as well. However, the variations of the more complex base model *II* presents more challenges to the RL agents. Only the variation with doubled process times yields a stable result across all 100 validation runs (Figure 5), while the remaining three variations yield outcomes that are less stable, in particular in the case with extremely mixed process times. In most cases deadlocks are observed, which are likely caused by multiple simultaneous machine outages. In the "Mixed Const" variation, the impact of these simultaneous outages tends to be greater than in variations with equally balanced process times. With balanced process times, the throughput reduction caused by an outage is unaffected by which machine of a work center has an outage, as the machines are identical. However, in the "Mixed Const" variation it does matter because with unequal process times more distinct state combinations may occur which increases the state-action spaces of the agents. Since the training was performed with balanced process times, the DQN agents were not sufficiently adapted to these rare state-action combinations. This phenomenon becomes more apparent in more complex systems, where the chance of simultaneous outages increases with the number of machines and work steps.

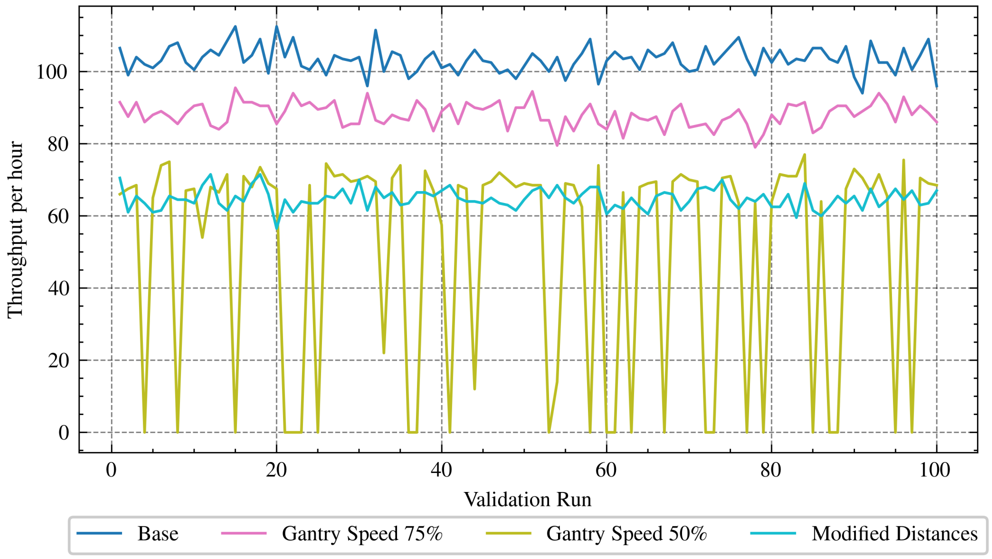

Figure 3: Results of validation runs for the base model *II* (3 H-style gantries; 5 process steps) with *gantry variations*.

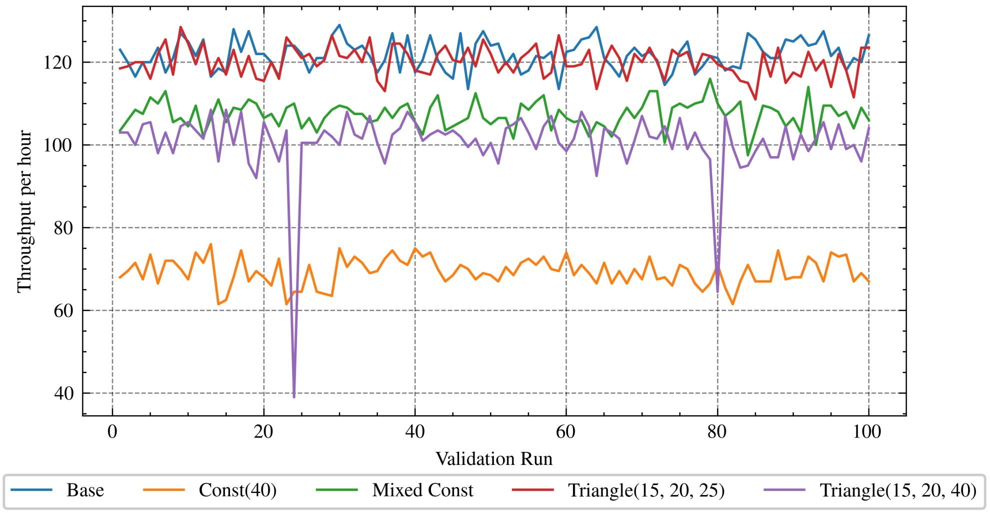

Figure 4: Results of validation runs for the base model *I* (2 H-style gantries; 3 process steps) with *process variations*.

However, despite their diminished stability, the average throughput of the variations Triangle $(15,20,25)$ and Triangle $(15,20,40)$ remains reasonable. As Table 4 shows, the agents are able to achieve an average throughput of 96.4 parts per hour for the Triangle $(15,20,25)$ variation. This is about 96% of the average throughput per hour achieved in the directly trained case. And even with an extremely skewed process time distribution, like Triangle $(15,20,40)$, the throughput achieved in the variation is still about 91% of the mean throughput per hour of the agent trained with the Triangle $(15,20,40)$ distribution directly.

Table 4: Mean throughput per hour with CI-95% for the validation runs of the *process variations* of the base model *II* and the directly trained modified model.

Variation	Base Model Agents	Directly Trained Agents
Mixed Const	58.9 [50.59, 67.21]	113.9 [113.18, 114.59]
Triangle (15, 20, 25)	96.4 [92.98, 99.79]	100.6 [99.82, 101.40]
Triangle (15, 20, 40)	72.9 [68.74, 77.02]	80.4 [78.64, 82.20]

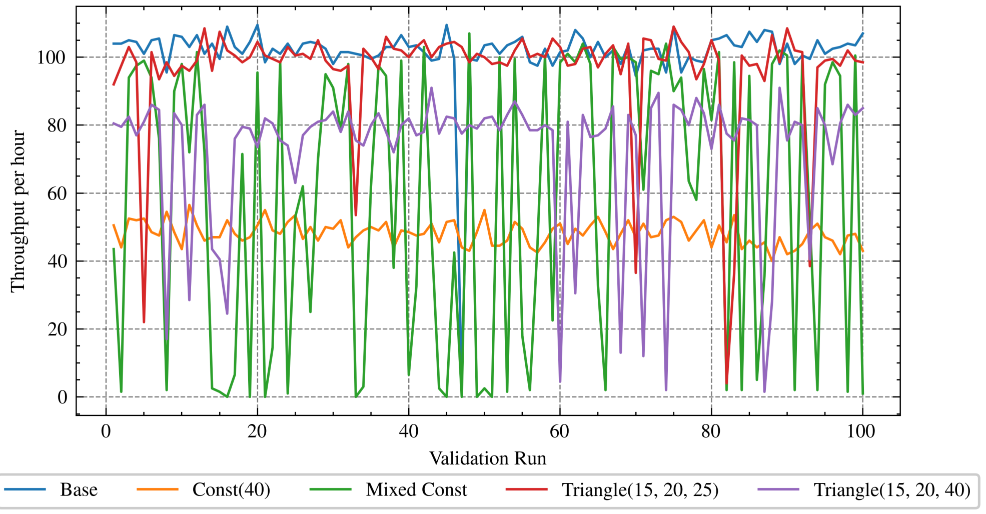

Figure 5: Results of validation runs for the base model *II* (3 H-style gantries; 5 process steps) with *process variations*.

5 CONCLUSION

This paper presents a MARL approach for the control of flow shops with gantry robots as automated material handling system. Contrary to the usual analysis of whether the agents are able to control the production system efficiently, the focus in this paper is on the analysis of their generalization capabilities. Generalization is an important aspect in Industry 4.0 scenarios, which require a certain degree of autonomous adaptability of the control systems to changing conditions on the shop floor. For the analysis, agents were trained for base models and then the models were modified with regard to the parameters concerning the gantry robots and the production process. Some extreme modifications were made in order to better demonstrate the limits of the agent's generalization characteristics. It can be shown, that in general the presented MARL agent is able to cope with most variations. For the base model *I*, the agent achieves stable controls with very few exceptions. Even in the edge cases, the efficiency of the agents in the variations is only about 10% below the efficiency of the agents that were trained directly on the modifications. For the more complex base model *II*, the results are similar, but not quite as good as in the first case. Here, the agent can also handle variations of the gantry robot system well, even in extreme cases. But especially for the variations of the process times, the results of the agent are not quite as stable as in the base model *I*, but still the throughput in the extreme variations do not fall below 90% of the throughput achieved in the direct training. In view of the sometimes very extreme and unrealistic variations, this is still considered to be a remarkable result.
In summary, the agents presented show good generalization properties within realistic limits. The authors consider it a possible approach for the control of flow shops with gantry robots in an Industry 4.0 scenario. Nevertheless, further research should be carried out to determine which training approaches can in principle be used to increase the generalization capabilities of the agents. On the other hand, agents that are trained

directly with the modifications will obviously show better efficiency in most cases. Therefore, it should be investigated how transfer learning approaches can help to retrain the agents quickly and efficiently, if there are major changes on the shop floor, which might exceed the limits of the agent's generalization capability.

AUTHOR BIOGRAPHIES

HORST ZISGEN is a full Professor in the Department of Mathematics and Natural Sciences at the Darmstadt University of Applied Sciences, Germany. His research interests include Monte Carlo methods, queuing theory, stochastic processes and Reinforcement Learning. He received a Diploma degree and a Ph.D. in Mathematics from the Clausthal Technical University. Before joining Darmstadt University of Applied Sciences he worked several years within IBM's research and development organization, where he held several technical leadership positions. His email address is horst.zisgen@h-da.de and his website is https://fbmn.h-da.de/zisgen-horst.

JANNIK F. HINRICHS is a Research Associate in the Department of Mathematics and Natural Sciences at the Darmstadt University of Applied Sciences, Germany. He received an M.S. degree in Data Science from the Darmstadt University of Applied Sciences. Prior to that, he earned a B.Eng. in Mechatronics and worked for several years as a software engineer at Gebr. Heller Maschinenfabrik GmbH, Nürtingen. His email address is jannik.hinrichs@h-da.de.