Exploring Scope of Using Multi-Agent Reinforcement Learning Systems for Efficient Warehouse Management with Robots

Ashish Rana, 1822317
ashish.rana@students.uni-mannheim.de

May 26, 2023

Contents

List of Tables

Abstract

In manufacturing, warehouse management is one of the laborious tasks especially in large scale warehouses for companies such as Amazon and Alibaba. Additionally, for optimization the moving goods and packages task is rapidly going through digital transformation by using mobile robots to build automated warehouses. This two-step technical problem of Multi-agent Pickup and Delivery (MAPD) requires the system to first assign tasks for agents and then generate collision-free paths. The more traditional optimization based approaches solve this problem by using different search and collision prevention strategies. The results from these studies, even though highly optimized, require large domain specific engineering efforts governed by warehouse specifications. Whereas, modern Multi-agent Reinforcement Learning Systems (MARLS) handles this two-step task with more flexibility by using approaches like, reward shaping, and hierarchical learning etc. In our study we qualitatively and quantitatively compare these approaches and address their corresponding advantages and limitations in detail. Also, the related code repository implementation and experimentation is present at github.com/ashishrana160796/analyzing-cooperative-marls.

1 Introduction

In automated warehouse management systems, ideally a team of robots works together to fulfill customer orders. This team of robots unlike humans can work tirelessly and increase throughput manifolds for any warehouse management organization. Coordinating robots in such a large system requires multiple components to work together like assigning management jobs to robots, finding optimal collision free paths, and infrastructure maintenance and upgradation etc. This complete task problem has been formally studied in the past as separate Multi-robot task assignment (TA) and Multi-agent Path Finding (MAPF) problems [14, 22]. But, these heuristic methods require significant engineering efforts and constant fine-tuning based on changing warehouse configurations and customer demand distribution. Additionally, approaches only targeting fully automated robotic solutions lack practicality in deployment considering high maintenance, unexpected downtime possibilities, and discarding already existing mature infrastructure involving humans. Considering the above described major systematic issues, in this manuscript we explore two MARLS experiments which attempt to introduce flexibility in operations, and utilization of existing warehouse management infrastructure for the warehouse management task.

In this manuscript, we first discuss some prior concepts for the automated warehouse management problem, and further elaborate on reinforcement learning terminologies for multi-agent scenarios. Second, we discuss experimentation and results for two MARLS studies, where the first study uses reward shaping to tackle sparsity problems in multi-agent warehouse settings. The second study focuses on developing a practical and scalable solution with hierarchical RL. It also benchmarks its proposed MARL approach performance against the industry standard heuristic approaches. Finally, we conclude our exploration study and practically quantify the MARL application performance with small experimentation [1].

2 Related Work

The MAPF problem is NP-hard to solve optimally where quality of solution is determined by makespan *(maximum of arrival times of all agents at goal locations)* or flowtime *(summation of arrival times of all agents at goal locations)*. Previously, studies have attempted to solve the warehouse management problem, where TA and MAPF are either solved separately or in a combined manner [14]. And, MAPF problem also has been studied in two different settings, where either the whole MAPF problem is solved or it is broken into smaller MAPF sub-problems for saving computations [22]. Integrating TA and MAPF is also explored as an improved alternative, where task assignment choices are calculated with real delivery costs instead of lower-bound estimates [5]. As an improvement for MAPF subtask, the corresponding study also uses marginal-cost assignment heuristic and large neighborhood search based meta-heuristic strategy for finding optimal paths. Further, studies have argued that MAPF independent of TA is a better alternative for building a more generalizable collision-free path finding system. The rolling horizon windowed MAPF solvers using traditional search algorithms for resolving collisions have also shown promising results [17]. But, the modularity with different components

[1] Experimentation code available at the repository github.com/ashishrana160796/analyzing-cooperative-marls

involved adds complexity to the pipeline, and makes the performance of the approach highly correlated with module design choices. Below, we further discuss different automated warehouse management paradigms, and relevant background for MARLS which can serve as an end-to-end alternative.

2.1 Practical Automated Warehouse Management Background

The order-picking task essentially consists of retrieving order items and delivering them to target location, it generally accounts for 55 % of operational cost for warehouses [9]. Automation systems like Dematic Multishuttle1, Autostore2, and KIVA use the *pick-to-picker* paradigm, where autonomous systems move items to pickers for dispatch to customers [25]. These systems are more costlier for warehouses with varying scale and flexible demands in comparison to the *picker-to-pick* paradigm, which accounts for 80% of warehouses in Western Europe [8]. In this paradigm the picker robots go to the item locations to directly retrieve and dispatch the item. More practical automated warehouse management studies formulate their solutions around these paradigms for better benchmarking with existing solutions.

There are several key challenges for managing warehouses realistically at large scale for improving key performance indicators like, pick rate, idle time, and distance traveled. For *picker-to-pick* paradigm heuristic approaches like *Follow Me* (FM), and *Pick, Don't Move* (PDM) have shown promising results. In FM, a cluster of AGVs are assigned to each picker to follow, and a traveling salesman problem (TSP) solution for all the AGVs is generated to determine order fulfillment sequence. It improves the efficiency by minimizing idle time for pickers, but might increase their travel time and distance traveled leading to more energy consumption. Whereas in PDM, pickers are allocated to zones in the warehouse, and AGVs are permitted to travel throughout the warehouse. The AGV packages are picked by pickers based on the AGV's current location and AGV's target location's proximity, where AGV travels through the order list using the TSP solution. Pickers movement is limited to zones but requires timely movement for loading and unloading AGVs. This approach minimizes the travel distance for pickers but might lead to under-utilization of pickers.

The heuristic strategies in general require repeated fine-tuning based on different use-cases. Depending on the customer and context, we might have to change many factors like, item clustering design, order priority mechanism, demand and supply logistics, labor and automation workforce conditions, and regulatory factors. This is an iterative and resource heavy process involving multi-party interactions leading to regular additional engineering effort for obtaining the most optimized solution. This suboptimal behavior issue transpires multi-faceted solutions exploring automated agent-policy solutions which address scalability and productionisation concerns.

Practically relevant solutions for the automated warehouse management problem address three main aspects, namely: i) pipeline scalability, ii) solution environment, and iii) productionisation. The scalable solutions is expected to handle magnitude and complexity of different dimensions of problems like, number of item locations $|L|$, order distribution denoted by Z, number of vehicle and picker workers represented by $|V| + |P|$ respectively. With increasing complexity, handling large numbers of agents centrally becomes harder as joint action space grows exponentially. This makes decentralized individual executor agents a preferred choice in MARLS. MARL algorithms require many interactions with the environment for learning, and initially learnt joint policies might be highly suboptimal for given warehouse configuration [20]. Therefore, a high performance simulation platform is important for the development and testing pipeline. Finally, a practical system should exist in an ecosystem of integrable tools either on-premise or on cloud which can help execute the code in a distributed manner. And, also debug model performance after observing any suboptimal behavior during the execution stage.

2.2 Reinforcement Learning Problem Background

RL tasks can be classified into model-free and model-based modeling types, where in model-free learning agents learn to get rewards with policy actions and need experience with the environment to learn. But, in model-based learning agents already know the reward and probability transition functions i.e. internal environmental models. For warehouse management problems it is better to opt for model-free learning because internal environment dynamics might vary based on scenarios, and modeling such dynamics to great details might be computationally unnecessary as well. Both value and policy based approximation methods have shown promising results for multiple RL tasks. But, architectures using

both these approximation approaches, like actor-critic models, are more capable of handling complex scenarios [23]. For multi-agent systems the Markovian assumption does not hold, there the Markovian Decision Process (MDP) modeling of the RL problem is dropped. And, the stochastic game setting is chosen where a combination of agent actions determines the next state and reward with partially observable environment characteristic [11].

MARL systems can be divided into three types based on task characteristics and the agent relationships. First, cooperative type where agents work together to achieve a common goal having a common reward model. And second it's opposite, a competitive setting where the reward system is modeled as a zero-sum game. The third type is the mixed one which is less restrictive, the warehouse management problem can be modeled as cooperative or mixed scenario MARL depending on the use-case. Additionally, for learning and execution the MARLS can again be divided into three types, namely: i) Centralized Training Centralized Execution (CTCE), ii) Centralized Training Decentralized Execution (CTDE), iii) Distributed Training Decentralized Execution (DTDE). In CTCE, centralized policy is learnt and used by the agents, during training agents influence each other which might make learning inefficient. The vice-versa is applicable for DTDE which involves independent learners, and both these approaches fail to scale because of the learning task complexity. In CTDE agents share the same learning model and experience with common goals, but agents perceive their interactions differently which results in different behaviors. For warehouse management problems in simpler scenarios CTCE is sufficient, and for scenarios requiring scalability CTDE is the preferred approach, both provide simplicity and efficiency for respective modeling requirements [24].

In simpler scenarios, for value based methods Deep Q-Networks (DQN) learns non-linear optimal action-value function following the Bellman Equation. Whereas Double Deep Q-Networks (DDQN) select action from current Q-Network and evaluate it using the old Q-Network to avoid instability in Q-Values while learning. Further, Independent Q-Learning (IQL) maximizes joint rewards by observing the local information. Where individual agents execute the Q-Learning algorithm and calculate its own loss based on local information [23, 24]. For warehouse management and other RL tasks, defining different reward distributions with changing configurations and use-case can be very hard. Further, not assigning dense reward signals might create sparsity problems as well. For the sparse reward problem, several approach alternatives have been explored, like reward shaping, transfer learning, imitation learning, curriculum learning, curiosity-driven learning, and hierarchical RL (RL) [12].

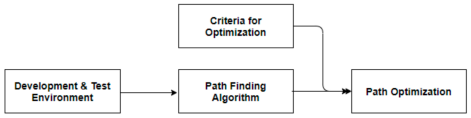
3 Multi-Agent Reinforcement Learning Approaches

In below subsections we analyze two different MARLS studies for automated warehouse management which analyze the warehouse package delivery problem at different abstraction levels. The first study presents a completely automated warehouse system with only robots, and handles theoretically novel sparse reward problems to improve existing RL algorithms [16]. Whereas, the second study envisions highly practical mobile robot and human picker based hybrid systems at more realistic scales [15]. This study further compares its findings and results with existing heuristic benchmarks to accurately quantify its performance in comparison to existing successful approaches.

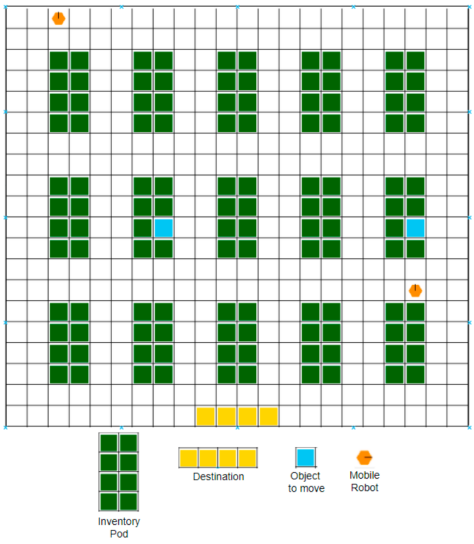
3.1 Reward Models on Segmented Actions for Agents

Modeling automated warehouse management as MARLS gives the solution relatively better flexibility and adaptability in comparison to traditional heuristic based approaches. But, with MARLS systems their own set of challenges needs to be resolved, like defining appropriate reward schemes. Also, reward engineering is especially challenging for MARLS and might lead to unexpected consequences if appropriate domain knowledge is missing [4]. For modeling this problem, only after a large sequence of actions within a complex delivery path a reward is assigned to the agents, which leads to the sparse reward problem. The experimentation study elaborated in this section resolves this issue by using a separate dual reward system during the initial and ending stages of the package delivery [16]. This newly defined reward system eventually helps in learning the package delivery task in a more stable manner for multiple robots in the warehouse environment.

The path planning task for delivering packages in the automated warehouse management problem comprises components like, path finding algorithm, path optimization, and optimization criteria for given development and testing environment, as shown in Figure 1a. The reinforcement learn-



(a) Path planning steps for optimization.



(b) Example of warehouse environment layout.

Figure 1: (a) Different modules in package delivery problem for the automated warehouse layout. (b) Warehouse grid layout with green inventory pods from which packages are delivered to yellow workstations. *(Image Credit: Lee et al.)*

ing approach for multiple agents abstracts away these heuristic specific components and attempts to model the whole process as an end-to-end task. Also, for the warehouse environment the reinforcement learning problem is developed in an abstracted and simplistic sparse simulation environment. The RWARE environment used in the dual reward system study simulates package shelf pick-ups by robots, workstation drops, and empty shelf returns in the specified layout presented by Figure 1b. In this environment, the mobile robot first finds and reaches the object location to pick up the object and place it on a transportation tray. After that, the robot places the package on the workstation and delivers the transportation tray back to initial position. For the experimentation the final workstation destination is fixed but robot starting and initial package locations are changed randomly.

The Figure 2 functionally describes the MARLS pipeline and the incorporation of the dual segmented reward model into that pipeline. The dual segmented reward model applies specific rewards for partial actions instead of full actions, where a full action is a union of sequence of partial actions. A partial action functionally defines a subtask only, like move to pick up an object, move to destination with object, return empty tray back etc. Whereas a full action comprises all the partial actions needed to deliver the object, and return the empty tray until the whole task is completed. For distinguishing goal achieving agents, cooperative and competitive relationships are also mixed for agents. Here, a relatively larger reward is assigned to agents that achieve the goal in comparison to agents that didn't complete the task. The dual segmented reward model study divides the reward signal of full actions defined functionally into distributed dense rewards over partial actions. Previously, studies have shown effectiveness of this approach for tasks like dexterous manipulation, and path finding [10, 21].

3.1.1 Experiments and Results

By exploiting task specific domain knowledge the handcrafted dual segmentation model assigns rewards differently with two reward schemes during initial and later stages of model training. In Figure 3a representing the first reward model, in beginning learning stages the initial partial actions are rewarded with higher rewards. For later learning stages the second reward model is utilized which encourages

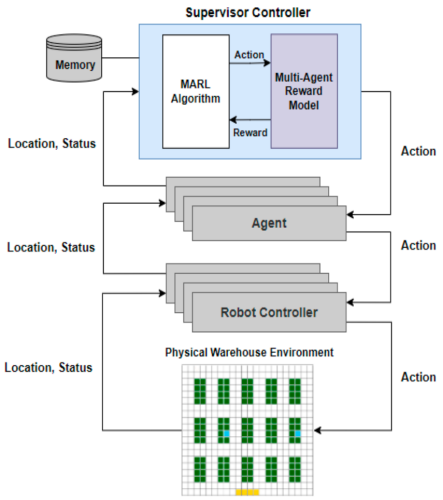
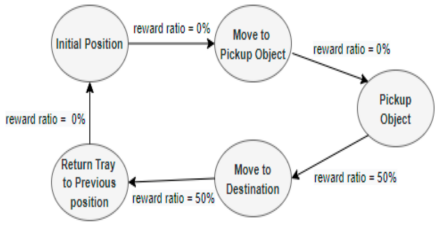
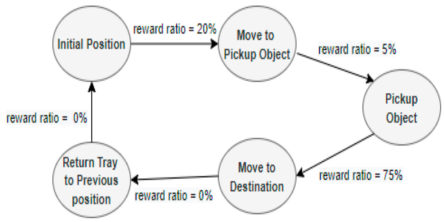


Figure 2: MARL mobile robot framework pipeline for warehouse management. *(Image Credit: Lee et al.)*



(a) First reward model for initial stages.



(b) Second reward model for final stages.

Figure 3: Segmented reward models assigning rewards to different subtask stages for the removing sparsity problem. *(Image Credit: Lee et al.)*

the task completion, refer Figure 3b. Additionally, for more stabilized learning a low learning rate of 0.00008, discount factor of 0.99, and reward decay rate of 0.99 with increasing episode count was also used. For experimentation, the environment scale includes 2 agents and 15 inventory pods where MARLS with DQN-based, and DDQN-based IQL algorithm models were trained. In IQL algorithms each agent maintains its own Q-values where it makes local information based belief updates, and assumes other agents are part of the environment. This approximation essentially results in the loss of convergence guarantee of the Q-Learning algorithm as other agents introduce non-stationarity in the environment.

In results, for baseline DQN and DDQN the instability is quite high in comparison to their dual segmented reward shaped trained counterpart agent systems which produces relatively less convergence variance. Additionally, from Figure 4 it is also observed that the dual segmented reward shaped models converged faster in comparison to models that weren't reward shaped. Also, at earlier stages the dual segmented reward shaped models demonstrate accumulation of positive reward signals which highlights higher initial exploration tendencies. From convergence curves, we also observe that both reward shaped DQN and DDQN models have similar and better performance in comparison to their baseline counterparts where DDQN performs relatively better than DQN. Further, as a limitation the correlation between environment simplicity and better performance was also highlighted for this highly use-case specific reward system.

3.2 Scalable Hierarchical Learning for Agents

This scalable MARLS study with automated guided vehicles (AGVs) and human co-workers focuses on a more practical collaboration based scenario for managing warehouse logistics. This study specifically focuses on maximizing the order-lines *(order comprises several order-lines)* per hour. This metric in

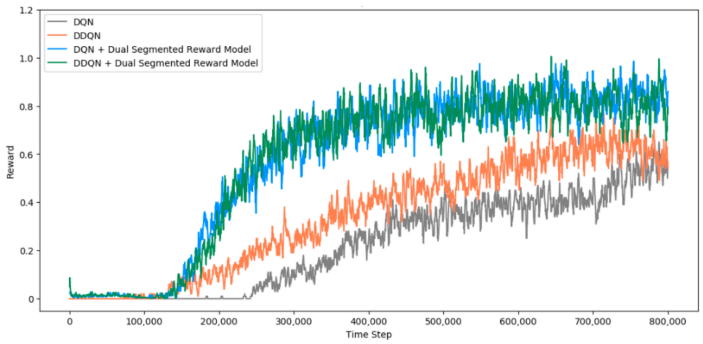


Figure 4: Reward accumulation analysis summary for DQN, DDQN, DQN with segmented reward model, and DDQN with segmented reward model. *(Image Credit: Lee et al.)*

turn also affects average distance traveled and idle time for AGV and human coworker agents. This study considers breaking down the complex action space by incorporating multi-layer hierarchy in Shared Experience Actor-Critic (SEAC) learning architecture i.e. analyzing decomposed action space via Hierarchical Shared Experience Network Actor-Critic (HSNAC) to observe performance gains. Also, Figure 5 highlights the solution pipeline where the MARL algorithm runs on the AI controller with on-premise deployment for lowering the downtime. The commands are transferred to AGVs via vehicle management system, and for human workers via a mobile device through an independent on-premise communication system.

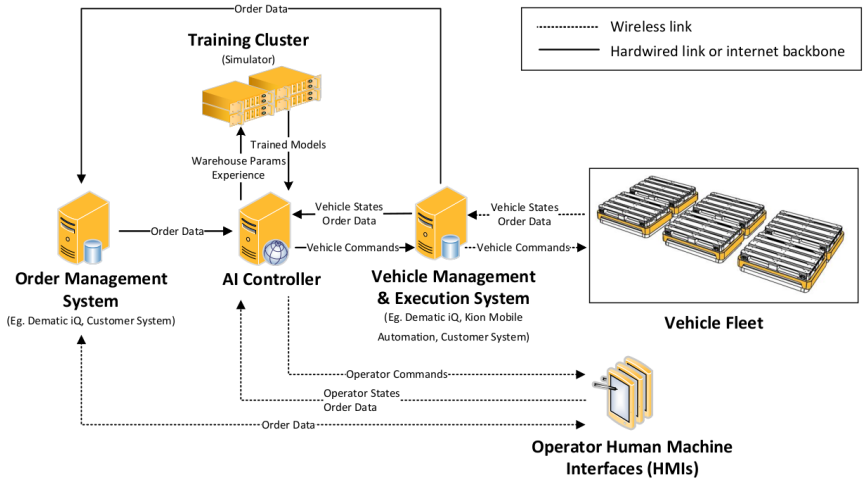


Figure 5: System pipeline for the proposed scalable MARL solution. *(Image Credit: Krnjaic et al.)*

The warehouse management problem previously was solved with three different strategies, namely: i) order-picking assistance by automated guided vehicles (AGVs), ii) Multi-Agent Pickup and Delivery Problem (MAPD), and iii) Multi-agent reinforcement learning (MARL). In the AGV-assisted order-picking model the order picking queueing problem quantifies the impact of zoning strategies to provide a polynomial time routing algorithm for simple warehouse designs [1]. This model is further refined with the introduction of disjoint zones and AGVs meeting the pickers at handover zones [28]. The MAPD paradigm formulates the task as a graph problem where agents and tasks are represented as nodes, and agents move between locations via the graph's edges. The objective is to minimize the time duration required for task completion and planning collision free paths [26, 19]. The scalable MARL system discussed here provides improvement on these strategies by avoiding usage of any hard constraints and heuristics on interaction amongst the agents. Other MARL systems have attempted to solve this problem with SEAC, and deep Q-networks [6, 13]. But, the scalable MARL approach uses Feudal Multi-Agent Hierarchies (FMH) from hierarchical RL (HRL) paradigm for decomposing action space and temporal abstraction [7, 2]. In FMH a manager agent uses individual agents to maximize

the environmentally-determined rewards for the warehouse management problem, which is here being formulated as a partially observable stochastic game (POSG).

Mathematically, the warehouse is defined by 3-tuples $\mathcal{W} = \{L,\ Z,\ W\}$. L represents two types of locations in the warehouse, namely: i) stored item locations ii) idle locations. Z defines warehouse and customer dependent order distribution, an order sampled from Z is defined by the tuple *(p, q)* representing order line and quantity respectively. Also, W represents workers like AGV *(V)* and human co-pickers *(P)*. An AGV is assigned an order z from distribution Z which comprises multiple *(p, q)* tuples is represented by z^v where $v \in V$. This study aims that for any given warehouse and order profile the joint policy π defined for workers W tries to maximize average pickup rate K, which is formally denoted by $\pi = \text{argmax}_\pi K(W, \pi)$.

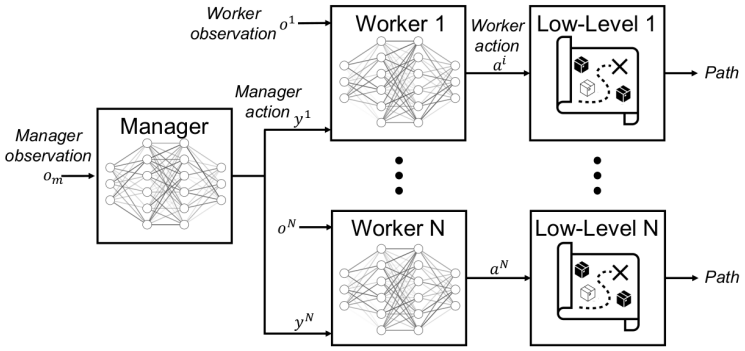


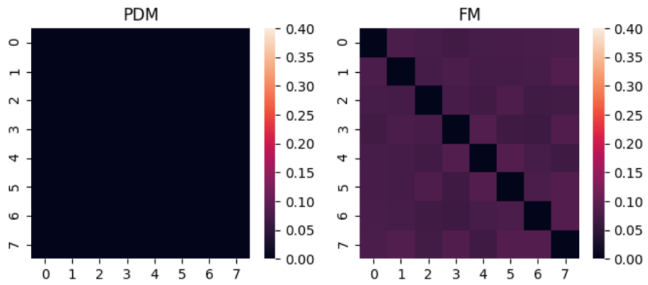
Figure 6: The Fedual Hierarchy architecture for the HSNAC MARL algorithm. *(Image Credit: Krnjaic et al.)*

The warehouse simulator implemented with OpenAI gym interface was aimed to be scalable and computationally tractable with low 0.2 FPS (frames per second) i.e. 5 seconds passes in game world with each game episode step [3]. This low FPS degrades the performance with lower action frequency to some extent. But, it simplifies model training with a reasonable enough time frame on which agent actions operate upon in the environment. For simplifications the collisions are not modeled in this environment, and the packages are also loaded automatically onto the AGVs without extra delays for additional quantity. In the system spawning mechanism, the number of agents remains constant, workers don't experience fatigue or failures, and the orders are assigned to AGVs in a first-in-first-out queue system.

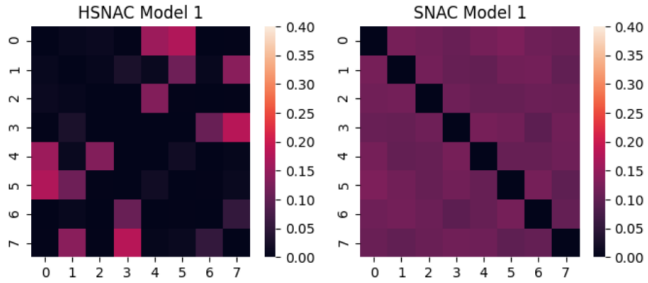
In this formulation, very uniquely agents select all their possible task target locations as their action space rather than usual directional AGV navigation movement actions. To supplement such actions, the Dijkstra's algorithm shortest paths are cached in the table where warehouse locations are represented in the graph. Significant memory resources are consumed for storing such path information, but it makes the path determination problem a hash-table lookup of *O(1)*. The distance calculations scales in complexity proportional to $O(|W||L|)$, a KD-Tree for decomposing graph coordinate space, and compiled vectorized function directly operational on machine level code was additionally used for optimality.

The MARL task interactions are formulated as POSG for N agents defined by seven dimensional tuple [11]. This tuple defines agent representations, state space, joint action space, partial observations, observation function, transition function, and reward function for all the agents in the system. And, like any other RL problem the aim is to learn a joint policy for all agents that maximize the discounted reward returns. The action space consists of mobile agents to visit all locations and pickers to visit all item locations. This results in large action space and action is also dependent on the agent's current and target locations. For observations, both AGVs and pickers observe current and target location of other agents. But, only pickers observe complete order lists whereas the AGV observe only their current order. Two different reward function schemes are followed for pickers and AGVs, first where the picker is rewarded 0.1 for completing the pick at timestep t, otherwise it is rewarded -0.05. Second, for each AGV 0.1 reward is given when either picked item is received or order is completed at timestep t, otherwise a reward of -0.05 is given.

The action space for each agent is really large, approximately being equal to the number of locations,



(a) Warehouse zoning heatmaps for heuristic approaches PDM and FM respectively.



(b) Warehouse zoning heatmaps for MARL approaches HSNAC and SNAC respectively.

Figure 7: (a) Heatmaps for picker-to-pick order line completion similarity data for PDM and FM approaches. (b) Equivalent heatmaps highlighting similarity between HSNAC and PDM, and SNAC and FM approaches respectively. *(Image Credit: Krnjaic et al.)*

and the actions will take different durations for terminations. This study employs FMH where the manager produces goals for the subordinates as shown in Figure 6. This concept is used to partition warehouse locations into sectors, where managers observe current and target locations for the agents to guide agents into their sector locations. After, the sector allocation agent's policy selects the next target, and then lower level controllers calculate the path to execute a sequence of actions for reaching the target. This sectoring strategy reduces the action space by a huge amount, and is further optimized by using action masking which filters out locations that are not part of the current order. The proposed sector zoning in FMH model is trained with the Shared Network Actor-Critic (SNAC) architecture [6]. The policy and value component networks are represented by artificial neural networks (ANNs) having 128 input dimensions with ReLU activations, and it outputs policy and value head for each agent. Each agent is parametrized with critic and value component networks components having 64 input dimensions with ReLU outputs.

3.2.1 Experiments and Results

For scalable experimentation testing, a large warehouse configuration of 1276 locations, 16 AGVs, 8 pickers, 22 partitioned zones, and 80 orders which comprises 5 order-lines on average was used in this study. Also, *pick rate* in order-lines per hour was used as the primary performance measure for all the experimentation approaches. This metric further expresses average frequency picks per episode values. The given HSNAC models were trained in PyTorch for 8000 episodes and 8 seeds. From Table 1, it is observed that the HSNAC approach generally outperforms all other approaches except for FM benchmark approach. This performance increase in pick rate is attributed to great reduction in idle time for AGVs and pickers but travel distance is increased to some extent. Qualitatively, multiple pickers competed for AGVs in experimentation which demonstrates emergence of competitive behavior. The approaches, PDM and FM utilize their pickers differently where PDM divides them in disjoint sectors, FM allows a more free movement. Additionally, it was observed that FM gives the lowest, and HSNAC gives the second lowest average episode length. A decrease in average episode

Table 1: Performance metrics *Distance (m)*, *Idle Time (s)*, and *Pick Rate* for HSNAC, SNAC, FM, and PDM approaches with ±95% confidence interval for AGVs and pickers.

MARL Methods	Metrics for AGVs			Metrics for Pickers		
	Distance	*Idle Time*	*Pick Rate*	*Distance*	*Idle Time*	*Pick Rate*
HSNAC	1887±3	779.48±1.91	69.99±0.10	2299±6	688.44±1.73	139.98±0.19
SNAC	1975±3	952.57±2.10	63.80±0.08	3678±9	585.99±1.28	127.59±0.16
Heuristic Methods	Metrics for AGVs			Metrics for Pickers		
	Distance	*Idle Time*	*Pick Rate*	*Distance*	*Idle Time*	*Pick Rate*
PDM	1529±4	1028.55±7.11	56.80±0.21	1104±5	1084.53±7.36	113.61±0.42
FM	1970±5	514.13±2.33	78.78±0.25	1620±6	582.49±2.40	157.57±0.49

length signifies efficient package delivery with reduced idle time. From Figure 8 it also demonstrated that HSNAC performs considerably well as compared to SNAC and PDM algorithms with higher pick rate throughput.

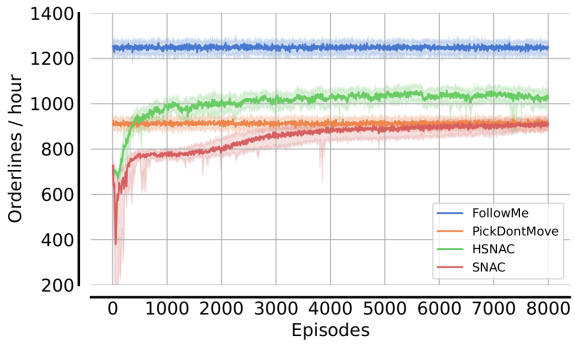


Figure 8: System pipeline for the proposed scalable MARL solution. *(Image Credit: Krnjaic et al.)*

For measuring picker utilization of the benchmarked algorithms, the individual picker's order-list completion location data is aggregated and normalized, and compared with cosine similarity measures of other pickers. From Figure 7a, we observe uniform similarity in FM, and no similarity with zoning strategy for PDM. Similarly with Figure 7b, we observe some extent of zoning in the HSNAC algorithm like the PDM approach, and uniform cosine similarity in the SNAC algorithm like the FM approach. It shows a structural sectoring or zoning emergence trend in the warehouse where picker regions seem to overlap with other pickers. The main disadvantage of using FM approach is direct dependence on performance of pickers which can lead to throughput dropouts if picker productivity drops. The major advantage of using HSNAC over the PDM industry standard is its flexibility i.e. it allows two or more pickers to serve secluded zones, and movement of pickers across zones as well.

4 Discussion

By exploring automated warehouse management approaches, for both heuristic and reinforcement learning paradigms we observe the high complexity associated with the task. [2] The heuristic approaches attempt to solve this problem as a modular problem, by optimizing each modular subproblem like TA and MAPF with traditional optimization and search based algorithms. These approaches as we observe have less flexibility and require more engineering effort when the problem involves changing parameters and modules leading to shift in optimization goal. Therefore, we further explore formulating this problem as a MARL problem, which leads to different challenges from theoretical ones like

[2]Bibliography papers available at the repository github.com/arana-initiatives/ai-portfolio-bibliography.

reward shaping to more practical ones like scalability. As part of this manuscript, we explored two studies which tackle these challenges and attempt to provide an end-to-end solution for the automated warehouse management problem.

For its merit, the experimentation study with dual segmented reward shaping model achieves good performance with very simplistic RL algorithms with their approach. But, still this study leaves out explanations of a plethora of experimentation factors in detail, and does not precisely quantify the performance improvements. First, the study mentions paradigms like, CTCE, CTDE etc. in their study but structurally does not explicitly give clarifications which paradigm their current approach belongs to or closely resembles. More importantly, the environment does not support concepts of full or partial actions but rather granular actions that give 2D planar directions to agents. Essentially meaning, that the reward model beholds more complexity which keeps track of subtask completion for each agent for reward distribution. In case it transforms action space into these functionally high level subtask actions, the problem complexity reduces drastically in comparison to the initial sparse problem. Additionally, these reward distribution magnitudes and their allocation are not discussed in precise detail in the study for each valid scenario. As *"Move to"* partial action's relative sequence in Figures 3a and 3b matters, if *"Move to Destination"* happens before *"Move to Object"* the proposed reward models will inconsistent. For example, as per the reward model specified in Figure 3b, after taking *"Move to Destination"* and *"Return Tray"* actions sequentially the partial reward will get assigned to the agent even if the object is not delivered. Hence, it is not made clear in the study whether the reward will only be assigned when specific complete reward sequences are traversed or partial reward with partial sequence will also be assigned. Therefore, without valid implementation details or repository there are several ambiguities for determining the true efficacy of the results produced in this study.

Second, for validating results only convergence plots were used to measure the efficacy. But, average episode length for task completion would give a more complete picture of the true model performance and learning capabilities. As agents in reward shaping formulation are often vulnerable to fall for positive reward cycles. And, it might be possible that agents are repeating redundant subtasks for additional rewards. This way agents are accumulating higher but redundant rewards which might not be necessary for the overall optimization process. The reward model designed in the study does the majority of the heavy lifting in the learning task. It does that qualitatively by breaking down the complex value function into a simpler value function alternative to enable learning with partial actions. Also, this model might only be tuned for this particular type of package delivery design in the warehouse, and possibly will not generalize well with some changes in the problem. Finally, the study mentions but does not compare its performance against the other stated relevant approaches for sparse learning, like curiosity drive learning, curriculum learning etc. The advantages listed for this study only operate on simpler scales with the current proposed architecture. But, realistically it can be effective for small warehouses practically in fully automated settings. Also, the utilization of the proposed dual segmented reward model might help other state of the art methods (SOTA) to achieve better performance as well.

The discussed hierarchical MARL study is highly scalable which is also compatible with existing real infrastructure and includes humans pickers as well. This approach is rather more practical as including humans can reduce unexpected warehouse downtime in case of major robotic infrastructure failure. Further this approach, unlike other MARL approaches, defines very high level actions with dense rewards which lead agents directly to the location, it further reduces the complexity drastically. And, this formulation closely resembles heuristic approaches as with this MARL formulation the agents select the location path from their actions directly. Here, unlike other standard MARL formulations, the agent does not iteratively learn to explore the environment with experience efficiently, but rather learns to select the shortest path in the given scenario. Storing such paths is a very computationally expensive step, and might introduce warehouse configuration consistency requirements, like heuristic approaches. It is better than heuristic approaches from the aspect that order demand distribution does not affect the system performance.

With the FMH design for complexity reduction this study fails to surpass FM which utilizes its agents more effectively. To its merit this study does outperform well established heuristic standard PDM. But, with higher average travel distance which practically would cost higher robotic maintenance and human welfare concerns. For realistic implementation the simplicity of the environment needs to be upgraded iteratively, as simulation results often cannot be directly translated to the real world

applications [27]. For example, realistically loading may take variable time depending on the required quantity, the collision-free assumption would not hold in real world, requiring agents to re-route dynamically. This study can further improve its results with addition of energy utilization penalization, and AGV agent communication to suppress the observed competitive behavior in the study. Also, warehouses do not operate in isolation but are part of a bigger supply-chain infrastructure, and the proposed solution should be adaptive of these changes [18]. Finally, the scalable solution provided in this study is very reasonable in terms of its MARL problem implementation and proposed enterprise design. And, can possibly be deployed easily into the real world after the improvement suggestions.

5 Conclusion

In conclusion, with our exploration we observe the efficacy of using MARL approaches for the warehouse management problem, and increased flexibility for providing end-to-end pipeline with minimal use-case specific engineering efforts. Second, we also explore different challenges associated with building scalable MARLS, and the engineering effort required for defining meaningful rewards for the same. Further, the relevance of heuristic based methods is also demonstrated in the scalable MARL study, where the RL problem's action space closely resembles heuristic problem formulation's search criteria. In our literature exploration, we couldn't find the recent SOTA MARL methods being compared with the heuristic benchmarks for this task. Also, literature exploring application of SOTA sparsity handling methodologies to mitigate the sparse rewards problem was also non-existent for this use-case. Even after these existing literature and experimentation limitations, we observe that MARL problem formulation and system design are capable of building scalable and human-centric solutions for manufacturing sectors.

References

[1] Azadeh, K., Roy, D., and De Koster, M. Dynamic human-robot collaborative picking strategies. *Available at SSRN 3585396* (2020).

[2] Barto, A. G., and Mahadevan, S. Recent advances in hierarchical reinforcement learning. *Discrete event dynamic systems 13*, 1-2 (2003), 41–77.

[3] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540* (2016).

[4] Cabi, S., Colmenarejo, S. G., Novikov, A., Konyushkova, K., Reed, S., Jeong, R., Zolna, K., Aytar, Y., Budden, D., Vecerik, M., et al. Scaling data-driven robotics with reward sketching and batch reinforcement learning. *arXiv preprint arXiv:1909.12200* (2019).

[5] Chen, Z., Alonso-Mora, J., Bai, X., Harabor, D. D., and Stuckey, P. J. Integrated task assignment and path planning for capacitated multi-agent pickup and delivery. *IEEE Robotics and Automation Letters 6*, 3 (2021), 5816–5823.

[6] Christianos, F., Schäfer, L., and Albrecht, S. Shared experience actor-critic for multi-agent reinforcement learning. *Advances in neural information processing systems 33* (2020), 10707–10717.

[7] Dayan, P., and Hinton, G. E. Feudal reinforcement learning. *Advances in neural information processing systems 5* (1992).

[8] De Koster, R., Le-Duc, T., and Roodbergen, K. J. Design and control of warehouse order picking: A literature review. *European journal of operational research 182*, 2 (2007), 481–501.

[9] Drury, J. Towards more efficient order picking. *IMM monograph 1*, 1 (1988), 1–69.

[10] Gudimella, A., Story, R., Shaker, M., Kong, R., Brown, M., Shnayder, V., and Campos, M. Deep reinforcement learning for dexterous manipulation with concept networks. *arXiv preprint arXiv:1709.06977* (2017).

[11] Hansen, E. A., Bernstein, D. S., and Zilberstein, S. Dynamic programming for partially observable stochastic games. In *AAAI* (2004), vol. 4, pp. 709–715.

[12] Huang, S., and Ontañón, S. Action guidance: Getting the best of sparse rewards and shaped rewards for real-time strategy games. *arXiv preprint arXiv:2010.03956* (2020).

[13] Kim, J.-B., Choi, H.-B., Hwang, G.-Y., Kim, K., Hong, Y.-G., and Han, Y.-H. Sortation control using multi-agent deep reinforcement learning in n-grid sortation system. *Sensors 20*, 12 (2020), 3401.

[14] Korsah, G. A., Stentz, A., and Dias, M. B. A comprehensive taxonomy for multi-robot task allocation. *The International Journal of Robotics Research 32*, 12 (2013), 1495–1512.

[15] Krnjaic, A., Thomas, J. D., Papoudakis, G., Schäfer, L., Börsting, P., and Albrecht, S. V. Scalable multi-agent reinforcement learning for warehouse logistics with robotic and human co-workers. *arXiv preprint arXiv:2212.11498* (2022).

[16] Lee, H., Hong, J., and Jeong, J. Marl-based dual reward model on segmented actions for multiple mobile robots in automated warehouse environment. *Applied Sciences 12*, 9 (2022), 4703.

[17] Li, J., Tinka, A., Kiesel, S., Durham, J. W., Kumar, T. S., and Koenig, S. Lifelong multi-agent path finding in large-scale warehouses. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2021), vol. 35, pp. 11272–11281.

[18] Lu, J., Liu, A., Dong, F., Gu, F., Gama, J., and Zhang, G. Learning under concept drift: A review. *IEEE transactions on knowledge and data engineering 31*, 12 (2018), 2346–2363.

[19] Ma, H., Hönig, W., Kumar, T. S., Ayanian, N., and Koenig, S. Lifelong path planning with kinematic constraints for multi-agent pickup and delivery. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2019), vol. 33, pp. 7651–7658.

[20] Papoudakis, G., Christianos, F., Schäfer, L., and Albrecht, S. V. Benchmarking multi-agent deep reinforcement learning algorithms in cooperative tasks. *arXiv preprint arXiv:2006.07869* (2020).

[21] Sartoretti, G., Kerr, J., Shi, Y., Wagner, G., Kumar, T. S., Koenig, S., and Choset, H. Primal: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters 4*, 3 (2019), 2378–2385.

[22] Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Kumar, T., et al. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search* (2019), vol. 10, pp. 151–158.

[23] Sutton, R. S., and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

[24] Wen, G., Fu, J., Dai, P., and Zhou, J. Dtde: A new cooperative multi-agent reinforcement learning framework. *The Innovation 2*, 4 (2021).

[25] Wurman, P. R., D'Andrea, R., and Mountz, M. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. *AI magazine 29*, 1 (2008), 9–9.

[26] Xu, Q., Li, J., Koenig, S., and Ma, H. Multi-goal multi-agent pickup and delivery. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (2022), IEEE, pp. 9964–9971.

[27] Zhao, W., Queralta, J. P., and Westerlund, T. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)* (2020), IEEE, pp. 737–744.

[28] Žulj, I., Salewski, H., Goeke, D., and Schneider, M. Order batching and batch sequencing in an amr-assisted picker-to-parts system. *European Journal of Operational Research 298*, 1 (2022), 182–201.