

# Multi-Agent Reinforcement Learning for Cooperative Warehouse Automation: QMIX Value Decomposition for Sparse-Reward Coordination

Price Allman, Lian Thang, Dre Simmons, Salmon Riaz

*Course: Artificial Intelligence,*

*Department of Computer Science, Oral Roberts University, Tulsa, OK, USA*

December 2025

## Preamble

This paper was written as part of the Artificial Intelligence course at Oral Roberts University. Its purpose is educational: to document the methodologies, analyses, and results of the project completed during the Fall semester. Although it follows academic standards, it is not a formal research publication.

## Abstract

We present a comparative study of multi-agent reinforcement learning (MARL) algorithms for cooperative warehouse robotics. We evaluate QMIX and IPPO on the Robotic Warehouse (RWARE) environment and a custom Unity 3D simulation. Our experiments reveal that QMIX’s value decomposition significantly outperforms independent learning approaches (achieving 3.25 mean return vs. 0.38 for advanced IPPO), but requires extensive hyperparameter tuning—particularly extended epsilon annealing (5M+ steps) for sparse reward discovery. We demonstrate successful deployment in Unity ML-Agents, achieving consistent package delivery after 1M training steps. While MARL shows promise for small-scale deployments (2-4 robots), significant scaling challenges remain. Code and analyses: [project documentation](#).

## 1 Introduction

Our research investigates the complex task of multiple autonomous agents learning to coordinate and deliver packages in warehouse environments—a problem requiring implicit communication, collision avoidance, and efficient task allocation without centralized control. Traditional warehouse automation relies on centralized planning systems that face scalability limitations; multi-agent reinforcement learning (MARL) offers an alternative through decentralized learned policies, but requires solving the credit assignment problem.

We compare MARL algorithms on warehouse coordination: QMIX [Rashid et al., 2018] (value decomposition), IPPO (independent learning), and MASAC (centralized critic). Our study progresses from MPE for validation to RWARE for warehouse evaluation, culminating in Unity 3D deployment where agents demonstrate learned package delivery behavior. QMIX emerged as the best performer after systematic comparison.

Our contributions: (1) hyperparameter analysis showing default configurations fail on sparse-reward warehouse tasks, (2) comparative evaluation across algorithms and scales, (3) Unity ML-Agents integration demonstrating sim-to-sim transfer with successful package delivery, and (4) identification of scaling challenges. Full experimental details and results are documented in our [Quarto documentation book](#).

## 2 Related Work

**Value Decomposition:** VDN [Sunehag et al., 2018] factorizes joint Q-values additively ( $Q_{tot} = \sum_i Q_i$ ), while QMIX [Rashid et al., 2018] uses hypernetwork-based mixing for monotonic relationships. Both follow Centralized Training with Decentralized Execution (CTDE), well-suited for warehouse robotics where deployed robots act on local observations.

**Independent Learning:** IPPO trains separate policies per agent, treating others as environment. Despite theoretical non-stationarity limitations, it shows surprising effectiveness in cooperative settings [Yu et al., 2022]. MASAC extends soft actor-critic with centralized critics and entropy regularization.

**RWARE Environment:** Papoudakis et al. [2021] provide a standardized warehouse benchmark with sparse rewards, partial observability, and configurable difficulty based on grid size and agent count.

## 3 Methods

### 3.1 Problem Formulation

We model the task as a Dec-POMDP where each agent  $i$  receives local observation  $o_i$ , selects action  $a_i \in \{\text{left, right, forward, load/unload, no-op}\}$ , and the team receives shared reward based on deliveries. Sparse rewards make credit assignment challenging.

### 3.2 QMIX Architecture

QMIX factorizes the joint action-value function:

$$Q_{tot}(\tau, \mathbf{a}) = f_{mix}(Q_1(\tau_1, a_1), \dots, Q_n(\tau_n, a_n); s) \quad (1)$$

where  $f_{mix}$  is a mixing network with non-negative weights ensuring monotonicity ( $\frac{\partial Q_{tot}}{\partial Q_i} \geq 0$ ). We use GRU-based agents (64 hidden units) with 2-layer hypernetwork mixers.

### 3.3 Training Configuration

Our optimized configuration differs substantially from defaults:

Table 1: Hyperparameter Configuration: Default vs Optimized

| Parameter           | Default   | Optimized  |
|---------------------|-----------|------------|
| Batch Size          | 32        | 256        |
| Buffer Size         | 5,000     | 200,000    |
| Epsilon Anneal Time | 50,000    | 5,000,000  |
| Training Steps      | 2,000,000 | 20,000,000 |
| Learning Rate       | 0.0005    | 0.0005     |

Extended epsilon annealing proved critical—rapid decay prevents agents from discovering sparse rewards. Larger replay buffers enable learning from diverse experiences.

### 3.4 Unity ML-Agents Integration

We developed a Unity environment mirroring RWARE with 3D visualization ([implementation details](#)): 3 agents, 6 discrete actions, 36-dimensional LIDAR-based observations, 200-step episodes. Training uses no-graphics mode with 50x time scaling.

## 4 Experiments and Results

### 4.1 Environment Progression

We validated on MPE before transitioning to RWARE ([transition analysis](#)). MASAC on MPE converged in 30,000 steps achieving 63% improvement over random baseline ([MPE results](#)); RWARE required 20M+ steps for comparable performance—a 600x difference in sample complexity. Default hyperparameters produced zero learning after 2M steps, underscoring how dense-reward benchmarks mislead about algorithm readiness for sparse warehouse tasks.

### 4.2 Algorithm Comparison

Table 2: Performance Comparison on RWARE Environments (Test Return)

| Algorithm       | tiny-2ag-v2 | small-4ag-v1 | Training Steps |
|-----------------|-------------|--------------|----------------|
| QMIX            | <b>3.25</b> | <b>2.10</b>  | 20-30M         |
| IPPO (Advanced) | 0.38        | —            | 5M             |
| IPPO (Vanilla)  | 0.13        | —            | 20M            |
| Random Baseline | 0.05        | 0.02         | N/A            |

QMIX (3.25) outperformed advanced IPPO (0.38) by 8.5x on tiny-2ag-v2 ([IPPO analysis](#)). Even vanilla IPPO with 4x longer training achieved only 0.13—25x lower than QMIX. Value decomposition provides substantial advantages for sparse-reward coordination where agents must learn to implicitly divide labor for package delivery ([QMIX learning curves](#)).

### 4.3 Scaling Analysis

Performance degrades with agent count while training requirements grow super-linearly:

Table 3: QMIX Scaling Results Across Environment Sizes

| Configuration | Agents | Test Return | Required Steps |
|---------------|--------|-------------|----------------|
| tiny-2ag-v2   | 2      | 3.25        | 20M            |
| small-4ag-v1  | 4      | 2.10        | 30M            |
| medium-6ag-v1 | 6      | 1.45        | 40M            |

Scaling from 2 to 6 agents requires 2x more training (20M to 40M steps) while performance drops 55%. The joint action space grows as  $|A|^n$ : 6 agents yield 15,625 joint actions versus 25 for 2 agents, suggesting hierarchical approaches may be needed at scale.

### 4.4 Unity Deployment Results

We validated QMIX on Windows Server 2022 (Intel Xeon E5-2680, 196GB RAM):

Table 4: Unity Training Results on Windows Server

| Metric           | 500K Steps | 1M Steps |
|------------------|------------|----------|
| Test Return Mean | 95.2       | 238.6    |
| Peak Return      | ~150       | 443      |
| Episode Length   | 200        | 200      |
| Training Time    | 3h 2min    | 6h 18min |

Console logs confirmed consistent package delivery. At 500K steps, agents showed functional but suboptimal behavior; by 1M steps, policies became nearly deterministic (return std < 0.01) with smooth navigation. Despite continuous physics challenges versus grid-based RWARE, core coordination behaviors transferred successfully.

## 5 Discussion

Our primary research goal was to investigate the complex task of multiple agents learning to communicate implicitly and coordinate package delivery—a fundamental challenge in warehouse automation. QMIX’s value decomposition approach proved effective for this coordination problem, enabling agents to learn complementary roles without explicit communication channels.

**Key Findings:** (1) Default configurations fail—extended epsilon annealing (5M+ steps) is essential for sparse reward discovery ([hyperparameter analysis](#)); (2) QMIX substantially outperforms independent learning (8.5x on RWARE), validating value decomposition for credit assignment ([algorithm comparison](#)); (3) scaling challenges remain significant—industrial deployments (50+ robots) will require hierarchical approaches ([scaling analysis](#)).

**Unity Deployment:** Our Unity experiments demonstrated that learned coordination transfers to 3D environments with continuous physics. On Windows Server, agents achieved 238.6 mean test return after 1M steps, with console logs confirming consistent package pickup and delivery behavior ([deployment results](#)). The transition from grid-based RWARE to continuous Unity revealed that core multi-agent coordination—implicit task allocation, collision avoidance, and sequential delivery strategies—generalizes across environment representations.

**Practitioner Recommendations:** Extend epsilon annealing to 5M+ steps (100x default), use large replay buffers (200K vs 5K default), and plan for 20M+ training steps. Recommended configuration: batch size 256, buffer 200K, learning rate 0.0005, GRU dimension 64 ([configuration details](#)).

**Limitations:** Simulation-only evaluation (no physical robots), testing limited to 2-6 agents, simplified RWARE task structure, and extended training requirements may be impractical for rapid deployment.

## 6 Further Research

Several directions could advance warehouse MARL systems:

**Hyperparameter Optimization:** Systematic exploration of buffer sizes (128–512 episodes), learning rate schedules, and network architectures. Our 32-episode buffer was relatively small; larger buffers may improve sample efficiency and stability. Automated hyperparameter search could accelerate configuration for new warehouse layouts.

**Environment Scaling:** Extending beyond 4-6 agents to industrial scales (50+ robots) requires hierarchical decomposition or task partitioning. Larger warehouse grids, dynamic obstacles, multi-floor layouts, and time-sensitive deliveries would better reflect real-world complexity.

**Algorithm Comparisons:** Benchmarking QPLEX, MAPPO, and MAVEN against QMIX on warehouse tasks would clarify when value decomposition provides advantages versus policy gradient methods. Communication-based approaches may help with larger agent teams.

**Sim-to-Real Transfer:** Bridging simulation success to physical deployment remains the critical gap. Domain randomization, robust perception, and safety constraints need investigation.

## 7 Conclusion

We demonstrated that QMIX substantially outperforms independent learning for sparse-reward warehouse coordination (8.5x improvement), but requires extensive hyperparameter tuning—particularly extended exploration schedules. Our Unity integration validates sim-to-sim transfer. Scaling to industrial deployments remains an open challenge requiring hierarchical approaches.

### Code and Data Availability

All code, trained models, and experimental results are available at:

- GitHub: <https://github.com/pallman14/MARL-QMIX-Warehouse-Robots>
- Documentation: <https://pallman14.github.io/MARL-QMIX-Warehouse-Robots/>

### AI Assistance Acknowledgment

This research utilized Large Language Models (Claude, ChatGPT) for writing assistance, code debugging guidance, and documentation preparation. All experimental design, algorithm selection, and research conclusions were made by the authors. Comprehensive AI usage details: [https://github.com/pallman14/MARL-QMIX-Warehouse-Robots/blob/main/AI\\_ASSISTANCE\\_DISCLOSURE.md](https://github.com/pallman14/MARL-QMIX-Warehouse-Robots/blob/main/AI_ASSISTANCE_DISCLOSURE.md)

## References

- Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., and Whiteson, S. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. *International Conference on Machine Learning*, 2018.
- Sunehag, P., Lever, G., Gruslys, A., et al. Value-Decomposition Networks for Cooperative Multi-Agent Learning. *International Conference on Autonomous Agents and Multiagent Systems*, 2018.
- Papoudakis, G., Christianos, F., Schäfer, L., and Albrecht, S.V. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. *Neural Information Processing Systems Track on Datasets and Benchmarks*, 2021.
- Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Baez, A., and Wu, Y. The Surprising Effectiveness of PPO in Cooperative Multi-Agent Games. *Neural Information Processing Systems*, 2022.