International Research Journal of Multidisciplinary Sciences ISSN:3107-930X(ONLINE)
VOL-1 ISSUE-10 Oct 2025 page:1-6

Graph-Based Reinforcement Learning for Warehouse Robotics

Anam Shariq Birla Public School, Doha Qatar anam.s.khan92@gmail.com

Abstract: The problem of multi agent navigation (MAN) in dense, dynamic environments like automated warehouses presents significant challenges, including the need for efficient pathfinding, collision avoidance, and deadlock resolution. Traditional methods often struggle with scalability and the complex, non stationary dynamics inherent in multi agent systems. Deep Reinforcement Learning (DRL) has emerged as a promising solution, but standard implementations can lack a structured understanding of the environment's spatial topology. This paper introduces GraphProx , a novel framework that synergizes Graph Neural Networks (GNNs) with Proximal Policy Optimization (PPO) to address these limitations. In GraphProx, the warehouse floor is represented as a dynamic graph, where nodes correspond to locations (e.g., shelves, charging stations, intersections) and edges represent traversable paths. GNNs process this graph to learn complex spatial relationships and agent dependencies, producing rich embeddings that capture the global state. These embeddings are then used as input to a PPO based policy network, which learns decentralized, cooperative navigation policies for each agent. Our experimental results in simulated warehouse environments demonstrate that GraphProx significantly outperforms both classical Multi Agent Path Finding (MAPF) algorithms like Conflict Based Search (CBS) and vanilla multi agent PPO in key metrics, including task completion rate, makespan, and the number of collisions. The framework exhibits superior scalability and a remarkable ability to learn implicit coordination strategies, leading to more efficient and robust warehouse automation.

Keywords: Multi Agent Reinforcement Learning, Graph Neural Networks, Proximal Policy Optimization, Warehouse Automation, Path Planning, Collision Avoidance.

1. Introduction

The global logistics and e commerce boom has intensified the demand for highly efficient automated warehouse systems. Central to this automation is the problem of coordinating fleets of autonomous mobile robots (AMRs) for tasks like order picking, restocking, and inventory transport [1, 2]. This multi agent navigation (MAN) problem is characterized by a shared, dynamic workspace where agents must navigate from start to goal locations without collisions while minimizing total operation time (makespan) [3].

Classical approaches to MAN often rely on Multi Agent Path Finding (MAPF) algorithms, such as Conflict Based Search (CBS) [4] and its variants [5, 6]. While optimal or bounded suboptimal, these algorithms can be computationally expensive and struggle to scale to hundreds of agents in real time dynamic environments [7]. Furthermore, they typically require replanning upon encountering unforeseen dynamic obstacles or agent failures, leading to inefficiencies.

Reinforcement Learning (RL), particularly Deep RL, offers a paradigm shift by enabling agents to learn policies through interaction with the environment [8]. Proximal Policy Optimization (PPO) has become a cornerstone algorithm in this domain due to its stability and strong performance [9]. In multi agent settings, a common approach is to train decentralized policies using a centralized critic, as seen in Multi Agent PPO (MAPPO) [10]. However, a critical limitation of standard neural network architectures (e.g., CNNs, MLPs) in this context is their difficulty in capturing the explicit relational structure and long range dependencies between agents and the static environment [11].

Graph Neural Networks (GNNs) have recently emerged as a powerful tool for reasoning about structured data [12]. By performing message passing between connected nodes, GNNs can aggregate information from a node's local neighborhood, effectively learning the topology of the graph [13]. This property is exceptionally well suited for warehouse navigation, where the floor plan can be naturally modeled as a graph, and agents can be seen as dynamic elements interacting with this graph.

This paper proposes GraphProx , a novel framework that integrates GNNs and PPO to create a powerful and scalable solution for multi agent warehouse navigation. The core innovation lies in using a GNN to encode the entire warehouse state—including static infrastructure and dynamic agent positions—into a latent representation. This graph structured representation is then leveraged by a PPO actor critic architecture to learn efficient, collision free, and cooperative navigation policies. We hypothesize that this explicit structural inductive bias will lead to faster learning, better final performance, and improved generalization compared to non graph based DRL methods.

2. Methodology

The GraphProx framework consists of three main components: (1) a Graph Based Environment Representation , (2) a GNN based State Encoder , and (3) a Multi Agent PPO Policy Learner .

2.1 Graph Based Environment Representation

The warehouse is modeled as a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$.

Nodes $\mathcal{V}$: Each node $v_i \in \mathcal{V}$ represents a discrete location in the warehouse, such as a shelf access point, a charging station, a packaging station, or a corridor intersection. Each node has features x_i, which can include its type, current occupancy status, and its grid coordinates.

Edges $\mathcal{E}$: An undirected edge $e_{ij} \in \mathcal{E}$ connects two nodes v_i and v_j if an agent can move directly between them. This explicitly encodes the traversable paths of the warehouse.

Dynamic Agent Representation: Agents are incorporated into the graph as special nodes or as additional features on the nodes they currently occupy. In our implementation, we add an "agent presence" feature to each node. The graph structure itself remains static, but the node features are dynamically updated at each time step to reflect the current positions of all agents.

2.2 GNN Based State Encoder

The dynamic graph $\mathcal{G}$ is processed by a GNN to generate a latent representation for each node, h_i. We employ a Message Passing Neural Network (MPNN) framework [14]. For each node, the GNN performs K layers of message passing:

1. **Message Function:** For each node v_i, a message $m_i^{(k)}$ is computed by aggregating information from its neighboring nodes $\mathcal{N}(i)$:

$$m_i^{(k)} = AGGREGATE^{(k)}(\{ h_j^{(k 1)} : j \in \mathcal{N}(i) \})$$

where $AGGREGATE$ can be a mean, sum, or max pooling operation [15].

2. Update Function: The node's representation is updated by combining its previous representation with the aggregated message:

$$h_i^{(k)} = UPDATE^{(k)}(h_i^{(k 1)}, m_i^{(k)})$$

where $UPDATE$ is typically a learnable function like a Gated Recurrent Unit (GRU) [16] or a simple linear layer with non linearity.

After \(K \) layers, the final node embedding \(h_i^{(K)} \) captures structural information within its \(K \) hop neighborhood. These node embeddings form a rich, spatially aware encoding of the entire warehouse state, which is used by the policy network.

2.3 Multi Agent PPO with a Centralized Graph Critic

We adopt a centralized training with decentralized execution (CTDE) paradigm [17], which is effective in multi agent PPO [10].

Actor (Policy): Each agent \(a \) has a decentralized policy \(\pi_a \). The policy for an agent located at node \(v_i \) takes its local observation \(o_a \) (which includes its own goal and the embeddings \(h_j \) of nodes within a certain perceptual radius) and produces a probability distribution over actions (e.g., move North, South, East, West, Wait). The actor network is an MLP.

Critic (Value Function): A single, centralized critic \(V(\mathcal{G}) \) estimates the expected global return from the current global state. The input to the critic is the entire graph with its learned node embeddings \(\{h_i^{(K)}\} \). A graph level readout function (e.g., global mean pooling) summarizes the node embeddings into a single graph level representation, which is then passed through an MLP to produce the value estimate [18].

Proximal Policy Optimization (PPO): The PPO Clip objective [9] is used to update both actor and critic networks. The centralized critic provides a stable baseline, reducing variance during policy gradient estimation. The GNN's ability to model the entire scene allows the critic to accurately assess the global value of a state, facilitating more effective credit assignment and coordinated policy learning.

The training process involves agents interacting with the warehouse simulator, collecting trajectories of experiences (states, actions, rewards), and using these to update the GNN encoder, actor, and critic networks jointly. A reward function is designed to encourage progress towards goals while penalizing collisions and unnecessary waiting.

3. Experimental Setup and Results

3.1 Simulation Environment

We developed a grid based warehouse simulator inspired by [19]. Scenarios of varying complexity (e.g., 20x20, 50x50 grids) were created, featuring shelves, wide aisles, and designated stations. Tasks involved agents being assigned random start goal pairs. We compared GraphProx against three baselines:

1. Conflict Based Search (CBS) [4]: A state of the art optimal MAPF algorithm.
2. Vanilla MAPPO [10]: A standard multi agent PPO using a CNN over the 2D grid as its state encoder.
3. Random Walk: A simple baseline where agents move randomly until they reach their goal.

3.2 Key Performance Metrics

Success Rate (%): The percentage of agents that reach their goal within a time limit.
Makespan: The total time taken for all agents to complete their tasks.
Collision Count: The total number of agent agent collisions.
Average Path Length: The average number of steps taken per agent.

3.3 Results and Analysis

The results, averaged over 100 episodes in a 40x40 warehouse with 30 agents, are summarized in Table 1.

Table 1: Performance Comparison in a 30 Agent Scenario

Method	Success Rate (%)	Makespan (steps)	Collision Count	Avg. Path Length

Random Walk	42.1 ± 5.2	>500 (timeout)	18.5 ± 3.1	N/A
CBS	100.0	285 ± 22	0.0	18.2 ± 1.5
Vanilla MAPPO	88.3 ± 4.1	410 ± 45	5.2 ± 1.8	21.8 ± 2.1
GraphProx	99.5 ± 0.5	305 ± 28	0.3 ± 0.1	19.1 ± 1.7

Comparison with MAPPO: GraphProx consistently and significantly outperformed Vanilla MAPPO across all metrics. The GNN's structural encoding allowed agents to learn more effective collision avoidance and deadlock breaking strategies, as evidenced by the near perfect success rate and drastically reduced collisions. This confirms the hypothesis that an explicit graph representation provides a critical inductive bias.

Comparison with CBS: While CBS achieved optimality in this manageable scenario, its computational time grew exponentially with the number of agents. In a scalability test with 100 agents, CBS failed to find a solution within a 5 minute window for 40% of the episodes, whereas GraphProx maintained a success rate of over 97% with sub second decision times per step. GraphProx provides a highly competitive, real time capable alternative to classical planners in large scale settings.

Qualitative Analysis: Trajectories generated by GraphProx exhibited emergent cooperative behaviors. Agents were observed forming temporary "lanes," giving way to others with closer goals, and avoiding congested areas proactively. The GNN critic effectively learned to assign a low value to states leading to potential future deadlocks, guiding the agents towards more globally efficient paths.

4. Conclusion and Future Work

This paper presented GraphProx, a novel framework that integrates Graph Neural Networks with Proximal Policy Optimization for multi agent warehouse navigation. By explicitly modeling the warehouse as a graph and using GNNs to capture spatial dependencies, GraphProx overcomes key limitations of both classical MAPF algorithms and non structured DRL approaches. Our results demonstrate its superior performance in terms of efficiency, success rate, and scalability, highlighting its potential for real world warehouse automation.

Future work will focus on several exciting directions. First, we plan to investigate heterogeneous graphs where agents are represented as distinct node types, allowing the model to handle different robot capabilities [20]. Second, we will explore the integration of attention mechanisms within the GNN (e.g., Graph Attention Networks [21]) to enable agents to focus on more critical parts of the environment or specific neighboring agents. Third, transferring the trained policy to physical robot fleets and dealing with real world noise and uncertainty presents a significant challenge that will be addressed through sim to real techniques [22]. Finally, extending GraphProx to include task assignment and dynamic goal generation [23] would create a truly end to end warehouse management system.

References

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

[2] Azadeh, K., De Koster, R., & Roy, D. (2019). Robotized and automated warehouse systems: Review and recent developments. Transportation Science , 53(4), 917 945.

[3] Stern, R., Sturtevant, N. R., Felner, A., Koenig, S., Ma, H., Walker, T. T., ... & Boyarski, E. (2019). Multi agent pathfinding: Definitions, variants, and benchmarks. Symposium on Combinatorial Search (SoCS) .

[4] Sharon, G., Stern, R., Felner, A., & Sturtevant, N. R. (2015). Conflict based search for optimal multi agent pathfinding. Artificial Intelligence , 219, 40 66.
[5] Li, J., Tinka, A., Kiesel, S., Durham, J. W., Kumar, T. K. S., & Koenig, S. (2021). Lifelong multi agent path finding in large scale warehouses. Proceedings of the AAAI Conference on Artificial Intelligence .
[6] Boyrasky, E., Felner, A., Stern, R., & Sharon, G. (2016). Don't split, try to work it out: Bypassing conflicts in multi agent pathfinding. ICAPS .
[7] Ma, H., & Koenig, S. (2016). Optimal target assignment and path finding for teams of agents. AAMAS .
[8] Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction . MIT press.
[9] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 .
[10] Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., & Wu, Y. (2022). The surprising effectiveness of ppo in cooperative multi agent games. Advances in Neural Information Processing Systems .
[11] Iqbal, S., & Sha, F. (2019). Actor attention critic for multi agent reinforcement learning. International Conference on Machine Learning .
[12] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems .
[13] Kipf, T. N., & Welling, M. (2016). Semi supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 .
[14] Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. International Conference on Machine Learning .
[15] Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in neural information processing systems .
[16] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder decoder for statistical machine translation. EMNLP .
[17] Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, P., & Mordatch, I. (2017). Multi agent actor critic for mixed cooperative competitive environments. Advances in neural information processing systems .
[18] Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez Gonzalez, A., Zambaldi, V., Malinowski, M., ... & Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261 .
[19] Cohen, L., Uras, T., & Koenig, S. (2016). Feasibility study: Using highways for bounded suboptimal multi agent path finding. Eighth Annual Symposium on Combinatorial Search .
[20] Wang, X., & Sandholm, T. (2021). Reinforcement learning in multi agent systems. Communications of the ACM , 64(1), 84 93.
[21] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., & Bengio, Y. (2017). Graph attention networks. arXiv preprint arXiv:1710.10903 .
[22] Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., ... & Vanhoucke, V. (2018). Sim to real: Learning agile locomotion for quadruped robots. Robotics: Science and Systems .
[23] Khamis, A., Hussein, A., & Elmogy, A. (2015). Multi robot task allocation: A review of the state of the art. Cooperative robots and sensor networks .

[24] Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., & Monfardini, G. (2008). The graph neural network model. IEEE Transactions on Neural Networks .
[25] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature , 521(7553), 436 444.
[26] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human level control through deep reinforcement learning. nature , 518(7540), 529 533.
[27] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. nature , 529(7587), 484 489.
[28] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems .
[29] Foerster, J., Nardelli, N., Farquhar, G., Afouras, T., Torr, P. H., Kohli, P., & Whiteson, S. (2017). Stabilising experience replay for deep multi agent reinforcement learning. International Conference on Machine Learning .

[30] Sunehag, P., Lever, G., Gruslys, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., ... & Team, G. M. (2017). Value decomposition networks for cooperative multi agent learning. AAMAS .
[31] Rashid, T., Samvelyan, M., Schroeder, C., Farquhar, G., Foerster, J., & Whiteson, S. (2018). Qmix: Monotonic value function factorisation for deep multi agent reinforcement learning. International Conference on Machine Learning .
[32] Hafner, D., Lillicrap, T., Ba, J., & Norouzi, M. (2020). Dream to control: Learning behaviors by latent imagination. ICLR .
[33] Jaderberg, M., Czarnecki, W. M., Dunning, I., Marris, L., Lever, G., Castaneda, A. G., ... & Team, G. M. (2019). Human level performance in 3D multiplayer games with population based reinforcement learning. Science . [34] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. arXiv preprint arXiv:1606.01540 .
[35] Todorov, E., Erez, T., & Tassa, Y. (2012). Mujoco: A physics engine for model based control. IROS .
[36] Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., ... & Batra, D. (2019). Habitat: A platform for embodied ai research. Proceedings of the IEEE/CVF International Conference on Computer Vision .
[37] Behbahani, F., Shiarlis, K., Chen, X., Kurin, V., Kaseva, J., Hofmann, K., & Whiteson, S. (2019). From internal to internal external memory: A novel approach for multi agent navigation. ICRA .
[38] Long, P., Fan, T., Liao, X., Liu, W., Zhang, H., & Pan, J. (2018). Towards optimally decentralized multi robot collision avoidance via deep reinforcement learning. ICRA .
[39] Everett, M., Chen, Y. F., & How, J. P. (2018). Collision avoidance in pedestrian rich environments with deep reinforcement learning. IEEE Access .
[40] Chen, Y. F., Liu, M., Everett, M., & How, J. P. (2017). Decentralized non communicating multiagent collision avoidance with deep reinforcement learning. ICRA .
[41] Li, Q., Gama, F., Ribeiro, A., & Prorok, A. (2020). Graph neural networks for decentralized multi robot path planning. IROS .
[42] Khan, A., Zhang, C., Atanasov, N., Karydis, K., Kumar, V., & Lee, D. D. (2021). Memory augmented reinforcement learning for image goal navigation. IROS .
[43] Das, A., Gervet, T., Romoff, J., Batra, D., Parikh, D., Rabbat, M., & Pineau, J. (2019). Tarmac: Targeted multi agent communication. International Conference on Machine Learning .
[44] Jiang, J., Dun, C., Huang, T., & Lu, Z. (2020). Graph convolutional reinforcement learning. ICLR .
[45] Zambaldi, V., Raposo, D., Santoro, A., Bapst, V., Li, Y., Babuschkin, I., ... & Battaglia, P. (2018). Relational deep reinforcement learning. arXiv preprint arXiv:1806.01830 .
[46] Sukhbaatar, S., Szlam, A., & Fergus, R. (2016). Learning multiagent communication with backpropagation. Advances in neural information processing systems .
[47] Peng, P., Yuan, Q., Wen, Y., Yang, Y., Tang, Z., Long, H., & Wang, J. (2017). Multiagent bidirectionally coordinated nets: Emergence of human level coordination in learning to play starcraft combat games. arXiv preprint arXiv:1703.10069 .
[48] Berner, C., Brockman, G., Chan, B., Cheung, V., Debiak, P., Dennison, C., ... & Zhang, S. (2019). Dota 2 with large scale deep reinforcement learning. arXiv preprint arXiv:1912.06680 .
[49] Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., ... & Silver, D. (2019). Grandmaster level in StarCraft II using multi agent reinforcement learning. Nature , 575(7782), 350 354.
[50] Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., ... & Silver, D. (2020). Mastering atari, go, chess and shogi by planning with a learned model. Nature , 588(7839), 604 609.