*Knowledge and Decision Systems with Applications*

www.kadsa.org
eISSN: xxx-xxx

# A Cooperative Heterogeneous Multi-Agent System Leveraging Deep Reinforcement Learning

Rahim Ullah[1], Muhammad Saeed[1], Wajid Ali *[2], Junaid Nazar[2], Fakhra Nazar[3]

[1] Department of Computer Science, COMSATS University Islamabad, Islamabad 45550, Pakistan
[2] Department of Computer Science, Air University Islamabad, Pakistan
[3] Faculty of Computer Science, University of Poonch Rawalakot, Azad Kashmir, Pakistan

**ARTICLE INFO**

***Article history:***
Received 26 February 2025
Received in revised form 25 March 2025
Accepted 27 March 2025
Available online 30 March 2025

***Keywords:***

Multi-Agents; Deep Reinforcement Learning; Heterogeneous-Agents

**ABSTRACT**

The agent plays a pivotal role in the multi-agent environment. Previous studies have primarily focused on homogeneous agent collaboration, but the approaches used in these studies cannot be directly applied to heterogeneous agent collaboration. There is a need for a model that supports collaboration among heterogeneous agents, as real-world systems consist of agents with diverse shapes, functions, and tasks. This study proposes a framework that uses visual and vector observations to facilitate collaboration among heterogeneous agents. The seaport environment is simulated for this study. Visual observations and vector observations are employed to interpret signals from other agents and calculate distance between the agents in the environment. Two types of agents are used: the crane agent, which lifts and lowers heavy materials between the ship and the designated location, and the lifter agent, which transports bulky materials from the seaport to the storeroom. These agents are heterogeneous due to differences in size, capabilities, and tasks. Separate neural networks are trained for each agent. The results show that the proposed model outperforms the base model in terms of mean reward, extrinsic reward, episode length, and value estimation. In terms of value loss and policy loss, the proposed model performs similarly to the base model.

## 1. Introduction

Due to the recent improvements in computing power, machine learning has become a significant field in Artificial Intelligence. Machine learning has three main types: Supervised learning, unsupervised learning, and reinforcement learning. Reinforcement learning is unique, unlike

* *Corresponding email:*
wajidali00258@gmail.com
https://www.doi.org/10.59543/kadsa.v1i.13931

supervised and unsupervised which work on labelled and unlabeled data. Instead, it is about an agent [1], [2] interacting with its environment. The agent observes the environment, takes an action, and receives a reward or penalty based on the action. The agent aims to maximize the total cumulative reward over time by learning the best sequence of actions. Fig. 1 shows the reinforcement learning framework. Numerous practical uses for reinforcement learning exist, such as gaming [3], robotics [4], autonomous driving [5], resource utilization [6] and many more. To process high-dimensional input data, such as images and raw sensor data, Deep Reinforcement Learning is well-suited for effectively managing these kinds of problems [7], where traditional reinforcement learning techniques may struggle. Deep Reinforcement Learning (DRL) is a subfield of machine learning and reinforcement learning that combines reinforcement learning techniques with deep neural networks to enable more sophisticated and complex learning tasks. The achievement of DRL in playing Go [8] and video games [9] is very impressive and shows it can make decisions even better than humans. This success has sparked significant interest in extending its application to real-world tasks.

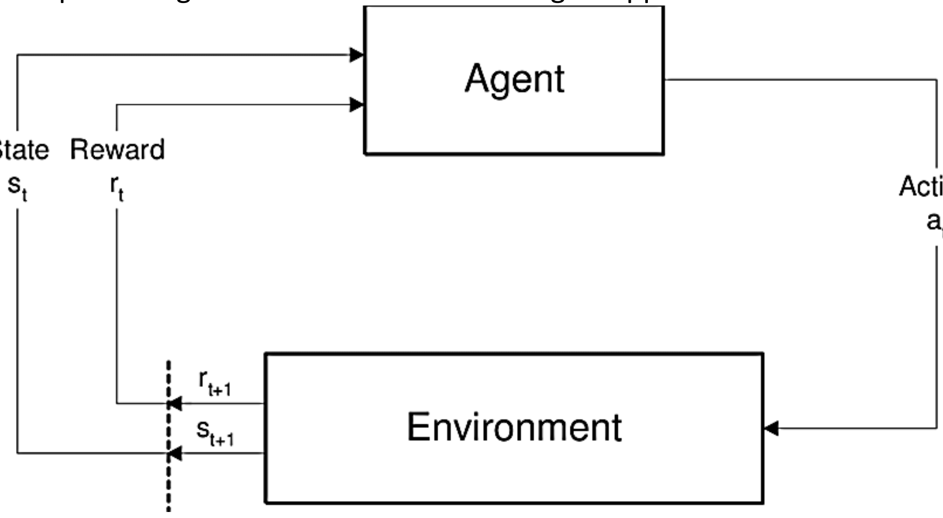

Fig. 1. Reinforcement learning framework.

In today’s era, deep reinforcement learning serves as a beacon of promise in the field of artificial intelligence. However, as the DRL focus moves toward solving real-world applications, the transformation from a single agent to a multi-agent system [10] becomes essential. Multi-agent system makes things more complicated because they need agents to work together, coordinate with each other and achieve common goals. While single-agent techniques have shown exceptional success, applying these theories to cooperative multi-agent systems presents impressive challenges [11]. The inherent challenges, such as managing the curse of dimensionality and multi-agent credit assignment, need innovative techniques designed for multi-agent systems.

Recent advancements in single-agent reinforcement learning systems have pushed the boundaries of what is achievable. However, moving to a multi-agent system demands a substantial shift [12]. The emphasis on effective communication and coordination in heterogeneous multi-agent environments underscores the necessity for Multi-Agent Deep Reinforcement Learning (MADRL). However, the existing MADRL approaches satisfy homogeneous agent environments, where all the agents have the same size, functionalities, and objectives. In many real-world scenarios, agents are heterogeneous [13], with diverse objectives and different sets of characteristics that create recent problems and need specialized solutions.

In this paper, we investigate the challenges and opportunities in Multi-Agent Deep Reinforcement Learning (MADRL), particularly focusing on the issues of non-stationarity and collaboration among heterogeneous agents. Agents cooperate and interact to conduct tasks in real-world situations, which presents unique challenges. Promising directions for future research in MADRL include addressing the problem of nonstationary and improving communication and collaboration among agents. Techniques for communication among homogeneous agents are not transferable to heterogeneous agents. The authors in [14] and [15] introduced MADRL methods for the collaboration of multiple agents, focusing on homogeneous agents expected to optimize a joint reward signal. However, in many real-world applications, agents are heterogeneous with varying objectives and environments [13]. The methods used in previous studies for homogeneous agents to solve these problems cannot be applied to heterogeneous agents [13]. Thus, there is a need to introduce MADRL methods to address non-stationarity and collaboration with heterogeneous agents. We propose a novel technique that effectively manages coordination and collaboration among heterogeneous multiagent systems, paving the way for advancement in real-world applications in multi-agent systems.

This paper presents a novel framework for facilitating collaboration among heterogeneous agents in multi-agent systems using deep reinforcement learning. It introduces an innovative approach that combines visual and vector observations with camera and ray perception sensors, enabling effective communication between agents of different sizes, capabilities, and tasks. The proposed model, evaluated in a simulated seaport environment, employs separate neural networks for each type of agent–crane and lifter–resulting in improved performance over the base model in terms of mean reward, episode length, value estimation, and extrinsic reward. The framework’s versatility is demonstrated through its potential applications in diverse real-world scenarios, such as shopping malls, seaports, airports, digital factories, and supermarkets.

This paper is organized as follows: Section II describes the detailed literature review, explaining the current research state and identifying the gaps, Section III provides a comprehensive proposed model, architecture, and implementation. Section IV describes the experimental result and evaluation of the proposed model and compares it with the base model. The last section, Section V talks about the conclusion of the paper and future work.

## 2. Literature Review:

There is a long history of ongoing research at the interface of reinforcement learning and multi-agent systems. Stone and Veloso [16] conducted one of the first studies in the field when they examined multi-agent systems from a machine learning standpoint and categorized the evaluated literature based on communication abilities and heterogeneous and homogeneous agent topologies. Due to the vast amount of network engagements, a variety of emerging technologies, including 5G networks, vehicular networks, unmanned aerial vehicle (UAV) networks, and the Internet of Things, will become diversified and decentralized. Each entity makes its own local decision in dynamic and unpredictable network situations to learn an optimal decision-making policy by interacting with unknown surroundings and to optimize network performance using learning algorithms such as single agent Reinforcement Learning (RL) or Deep Reinforcement Learning (DRL) [17]. However, such an approach may result in non-stationarity and is unable to capture network entity competition or cooperation. By enabling each network entity to determine its optimal policy, Multi-Agent Reinforcement Learning (MARL) improves the learning efficiency of network entities and addresses a range of network development challenges. In a recent study [18], DDMAC (Dynamic Data driven Maintenance Action Control) is applied to determine optimal system-level Inspection and Maintenance policies for systems with interconnected fatigue deterioration processes among components. A survey on MARL applications in the upcoming Internet generation was suggested by the authors of [19]. Specifically, they presented single-agent RL, and then discussed how MARL addresses several issues that would come up on the Internet in the future. The difficulties include packet routing, content caching, network access, transmit power control, computer offloading, trajectory design for UAV assisted networks, and network security concerns.

Deep reinforcement learning has been extremely useful in complex single-agent tasks in recent years, and it has just recently been applied to multi-agent domains [20]. For agents to effectively develop coordination, they require knowledge regarding the significance of ambient items to both them and other agents. For MARL, this is the most common challenge. A method called MAGNet (Multi-Agent Graph Network) was presented by the authors [21] for acquiring pertinent data in the form of a relevant graph. There are two phases to this method. The first stage involves learning a relevance graph. The actor-critical reinforcement learning network, which makes decisions for the agent and integrates message transmission methods, receives the relevance graph and associated state information in the second stage. They employed the well-known Pommerman game, which is frequently used to assess multi-agent systems, and the synthetic predator-prey game to evaluate the MAGNet approach. The performance of their method was noticeably higher than that of the most advanced MARL methods. Additionally, they show how well message forwarding, graph sharing, and self-attention work. In [22] suggests a robust POMDP strategy that takes model uncertainty into account. A technique known as the Deep Deterministic Policy Gradient (DDPG) is used in [23] to optimize quality inspection and maintenance in connection to industrial networks.

In multi-agent reinforcement learning, to maximize cooperative policies and manage unstable learning problems, agents need to learn communication protocols. Current systems based on actor-critic networks address agent communication [24]. It is difficult to comprehend the dynamic and non-stationary environment when the policies of other agents alter, though. Learning robust policies and improving sample efficiency are challenges for these methods. They offer a technique that considers agent communication when learning cooperative norms in multi-agent systems. To centrally train decentralized policies, the suggested method makes use of deterministic policy gradients and actor-critic networks based on recurrent neural networks [25]. Actor networks allow agents to decide what to do next and to communicate with each other via forward and backward pathways. With the help of gradient signals based on each actor network's contribution to the global reward, the critic network helps the actor networks learn. It is suggested to utilize auxiliary prediction networks to approximate state transitions and the reward function to tackle partial observe ability and unstable learning difficulties. By contrasting it with existing multi-agent reinforcement learning techniques in terms of learning efficiency and goal attainment during the test phase, they were able to show the usefulness and superiority of the novel strategy. It appears from the data that the suggested approach performed better than the substitute.

Multi-agent motion planning research now in existence can be categorized into two types: decentralized and centralized approaches [26] (where the objective is to guide every agent to its intended destination while providing information about each agent' s position, velocity, and target location). The split of computing effort across multiple agents is what makes decentralized systems scalable. In centralized systems, the number of agents determines how much they can scale. As a solution to the multi-agent motion planning problem, the authors suggested a novel goal-distance-based proxy reward for the GA3C-CADRL approach. This algorithm uses a more cautious approach, which reduces the number of collisions [27]. To address distributed motion planning issues in dynamic, crowded environments, they suggested combining deep reinforcement learning (DRL) with force-based motion planning (FMP) in a hybrid manner. According to simulations, their suggested solution outperforms DRL by 50% and requires up to 75% less extra time to complete an objective than FMP.

## 3. **Proposed Model:**

Previous research focuses on homogeneous multiagent systems (MAS). However, real-world applications often require heterogeneous multi-agent systems. The collaboration between homogeneous and heterogeneous agents differs due to variations in size, capabilities, and architecture. Available methods for homogeneous agents are inapplicable to heterogeneous agents, necessitating a novel approach. This research presents a framework for collaboration between heterogeneous agents, leveraging visual and vector observations combined with camera and ray perception sensors. These components do not require any specific size or architecture but enable effective communication among heterogeneous agents. Agents broadcast visual signals, which receiver agents observe using cameras or sensors. This method is applicable to all kinds of heterogeneous agent scenarios in the real world.

To tackle the communication issue among heterogeneous agents, this study employs visual and vector observations. Visual observations capture signals from other agents and are used to detect locations for tasks like dropping and picking up heavy material. To enable coordinated operations, every agent has a camera component that allows it to see its surroundings and decipher signals from other agents. Vector observations are numerical and commonly used for computing distances between agents and obstacles and as well as determining directions to specific points, making them particularly useful in environments with heterogeneous agents. Proximal Policy Optimization (PPO) is a gradient-based algorithm optimized using the Adam optimizer. It employs a three-layered convolutional neural network (CNN) with a ReLU activation function to process observations and determine actions. The neural network has three hidden layers, each with 256 units, and an output layer designed to correspond to the agent’s specific action set. For example, the lifter agent has four actions (left, right, forward, backward), while the crane agent can perform six actions (up, down, left, right, drop, pick). The Adam optimizer is used to optimize action values, and the ReLU activation function accelerates training convergence. PPO’s scalability and effectiveness make it a popular choice for various deep reinforcement learning applications.

The proposed system model consists of two agents: a crane agent and a lifter agent. A lifter agent observes the environment and acts and until it reaches the crane agent, the crane agent observes, at which point it broadcasts a visual signal. This broadcasted signal then observes the crane agent, which acts and picks up heavy material and keeps it on the lifter agent. The crane agent also broadcasts a visual signal and informs the lifter agent to move toward the storage area and drop off the material. This process continues until all the episodes are completed. The overall system model is demonstrated in Fig. 2, and the detailed interaction between agents is illustrated in Fig. 3.

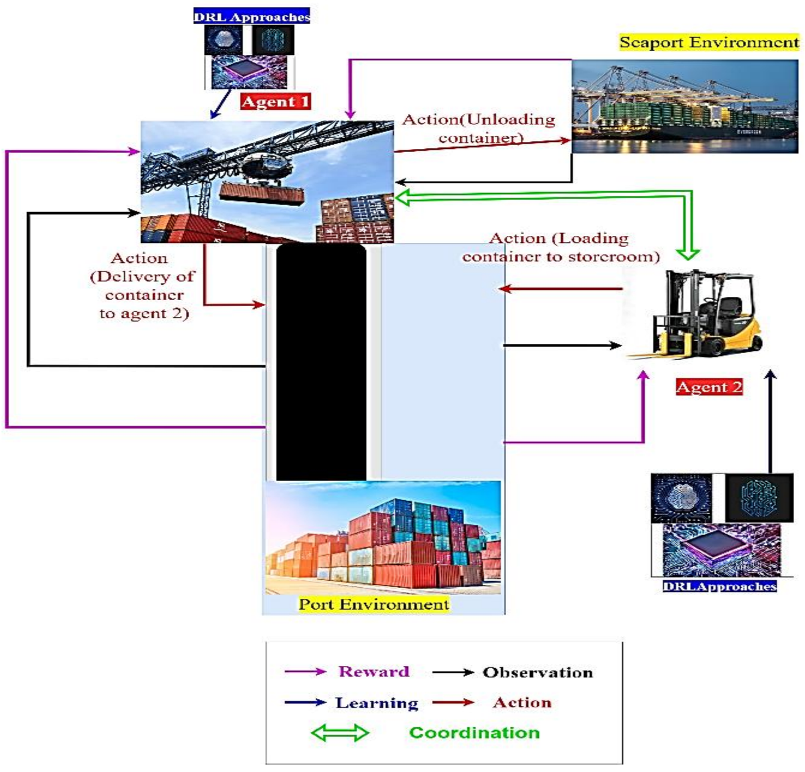

Fig. 2. System model 1.

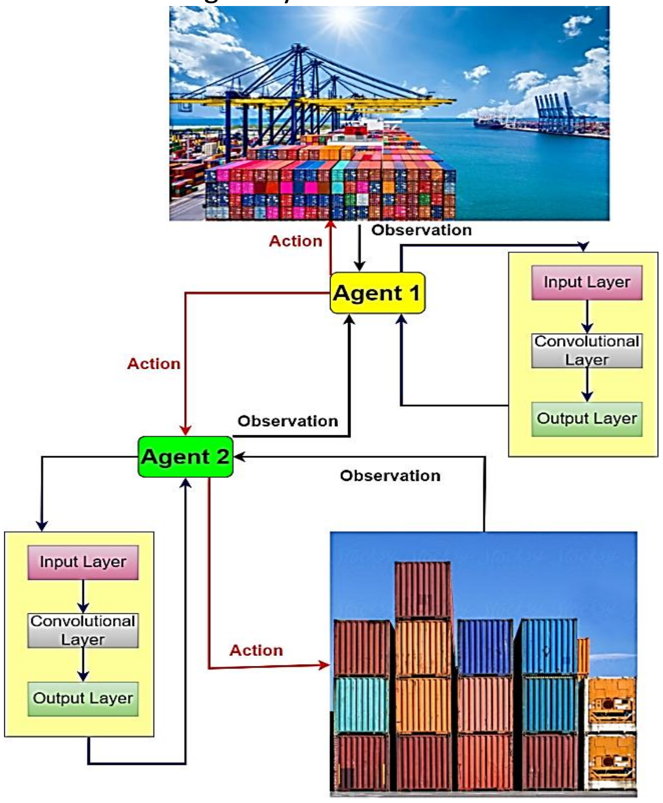

Fig. 3. System model 2.

Different components of the proposed system are described here:

- **Learning Environment:** The learning environment is created in Unity 3D and simulates the seaport with moving objects and buildings making it realistic and dynamic. The learning environment is 500 x 500 meters with agents equipped with ray perception sensors for vector observations and a camera for visual observations. For enhancing the realism in the environment Unity 3D rigid body is used. ML Agent toolkit is used for the implementation of reinforcement learning. It includes components like Brain, Academy and Agents, which are essential for training and controlling the agents in the Unity environment. Fig. 4 shows the learning environment.

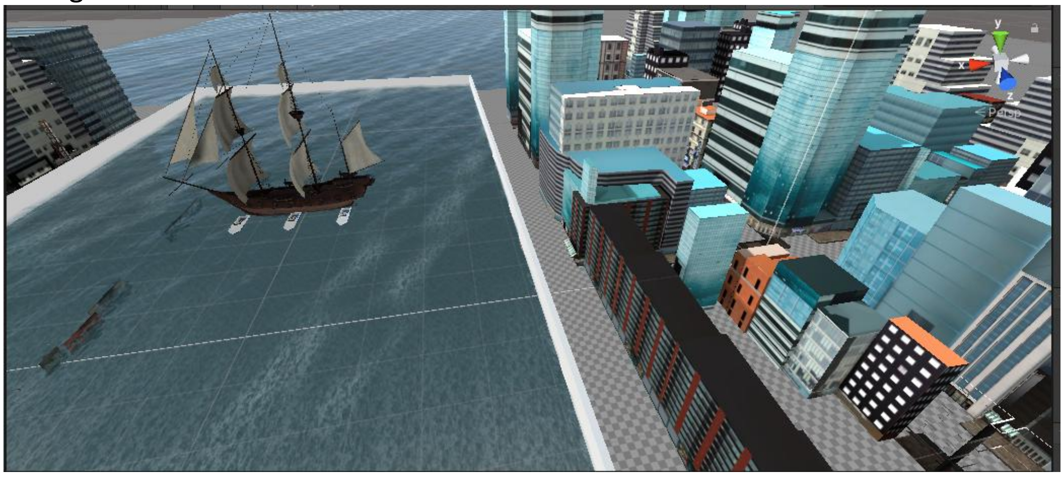
Fig. 4. Learning Environment.

- **Agent:** Crane and Lifter agents are used in the proposed environment. The Crane agent picks the materials from the seaport and places them on the lifter agent and then the lifter agent navigates and moves toward the storage area, where it drops off the materials. These actions are managed through visual and vector observations.

The implementation of the proposed model consists of using Unity 3D to create the learning environment and ML Agent toolkit for facilitating reinforcement learning. The training process uses episodic procedures, with 10,000 time-steps in each episode. Parameters such as batch size, buffer size, number of layers, and hidden units are optimized to ensure effective learning. Table I shows the respective hyper parameters and their corresponding values. If the lifter agent collides with obstacles, the episode ends too soon and is followed by fresh episodes with different actions and policies. The decision requester component serves as the agent’s brain, collecting observations and making decisions based on PPO algorithm neural network outputs. For successful actions such as correctly picking and placing the heavy material, a positive incentive is awarded; otherwise, the agent will be penalized.

TABLE I
TRAINING PARAMETERS

| **Parameter** | **Value** |
|---|---|
| Batch Size | 2048 |
| Buffer Size | 16384 |
| Hidden Layers | 3 |
| Hidden Units | 256 |
| Max Timesteps | 15000000 |

Algorithm 1 outlines the pseudo-code and working mechanism for the proposed model.
Algorithm 1 Pseudo code of proposed model

```
1: while episodes do not end do
        2: while Time steps do not end do
```

```
            3: while Distance of crane $\geq$ 0 do
            4: Lifter agent collects observations
        5: Select action
        6: Broadcast visual signal (Lifter)
        7: while Heavy material is not placed do
                8: Crane agent collects observations
                9: Act
                10: Broadcast visual signal (Crane)
                11: end while
                12: end while
        13: while Distance of store = 0 do
        14: Lifter agent collects observations
        15: Take action
        16: Drop heavy material
17: end while
18: end while
19: end while
```

Reward structures are important for agent performance. Crane agents get positive rewards for correctly picking heavy material or correctly placing heavy material and successfully interpreting signals. If the crane agent places the heavy material in the wrong place or drops away the heavy material from the lifter agent. The crane agent would get a heavy penalty as shown in Table II, because the safety of the other agents and objects is also very crucial in the environment. Equation 1 below is used to compute the total reward for the crane agent.

$rc = (rplace + rpick + rsignal) - (rdrop + raway)$ (1)

TABLE II
REWARDS GIVEN TO CRANE AGENT

| **Event** | **Reward** |
|---|---|
| Smoothly picking the heavy materials | 0.5 |
| Correctly placing the heavy material on the lifter agent | 0.5 |
| Incorrectly placing the heavy material on the lifter agent | -0.7 |
| Heavy material dropped away from the lifter | -0.7 |

Similarly, the lifter agent gets a positive reward, if it reaches to destination store or if it reaches to the crane agent after dropping the heavy material at the storage area or if it gives a correct visual signal. If the lifter agent collides with an obstacle or building, it will get a negative reward as shown in Table III. Equation 2 below is used to calculate the total cumulative reward for the lifter agent.

$rl = (rc + rs + rsignal) - (rb + ro)$ (2)

TABLE III
REWARDS GIVEN TO LIFTER AGENT

| **Event** | **Reward** |
|---|---|
| Agent reaches to destination store | 0.5 |
| Reaches to the crane agent after agent hits with some. Obstacle | -0.3 |

| | |
|---|---|
| Collides with some building | -0.3 |
| Agent gives correct visual signal | 0.5 |

The proposed framework is applicable in various real-world scenarios, including shopping malls, seaports, airports, digital factories, and supermarkets. Its ability to facilitate communication between heterogeneous agents makes it versatile and valuable across multiple domains.

## 4. EXPERIMENTAL RESULTS AND EVALUATION

This section describes the outcomes of the proposed model and the parameters that were assessed throughout the study.

The mean reward is the average of the rewards given to agents in each episode. This metric demonstrates how well the proposed model performs during the training. A higher mean reward means the agents have received sufficient training. Fig. 5 indicates that the proposed model outperforms the base model, with agents trained under the proposed model getting the maximum average reward of 0.02 compared to -0.28 for the base model.

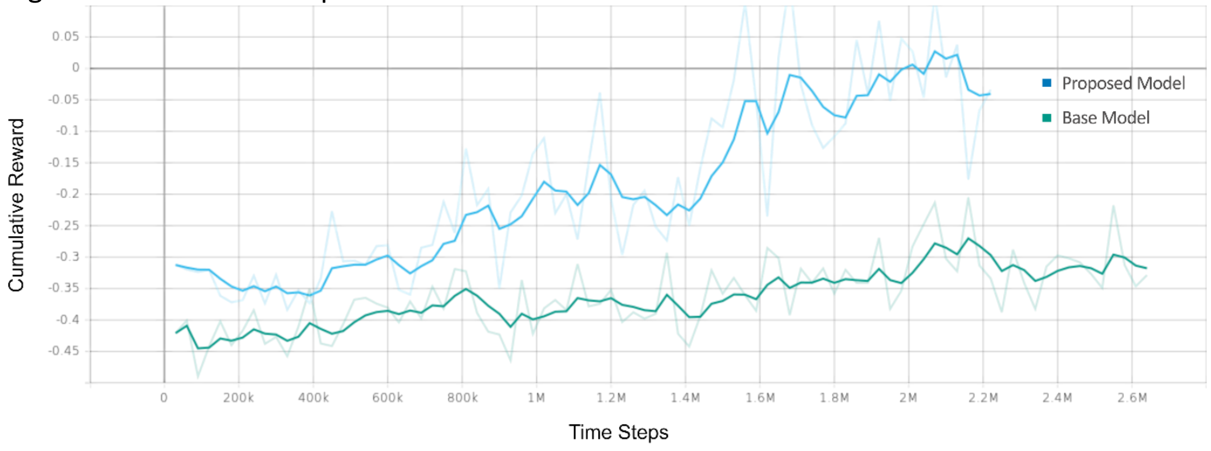

Fig. 5. Mean Reward.

Episode length is the number of time-steps an agent executes its task without colliding with any other object or failing. It indicates how long the agent can perform their task without making any mistakes. Agents trained with the proposed had longer episodes of 400 and the agents trained with the base model had episodes of 350 as shown in Fig. 6. It demonstrates the proposed model has a longer duration to complete its task without colliding with anything compared to the base model.

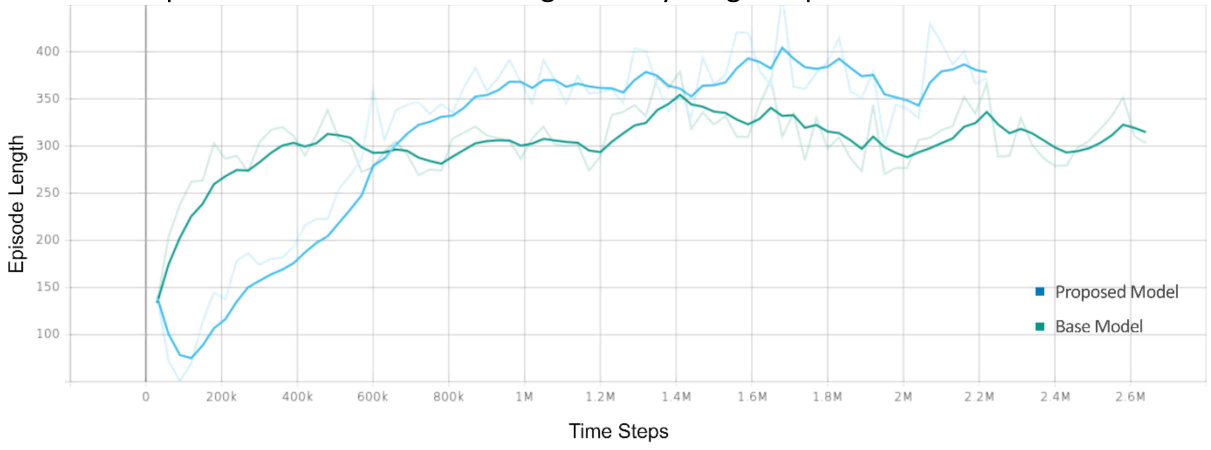

Fig. 6. Episode length.

Policy loss means, how the agent precisely learns the policy of the expert or the predefined set of policies. At the start, the agents have high policy loss, with the passage of time they learn the policy well and find the optimal policy. Fig. 7 shows the graph value decreases over time; it learns and adopts the policy and does not violate the predefined policy or expert policy. Both the proposed model and base model perform similarly in this metric.

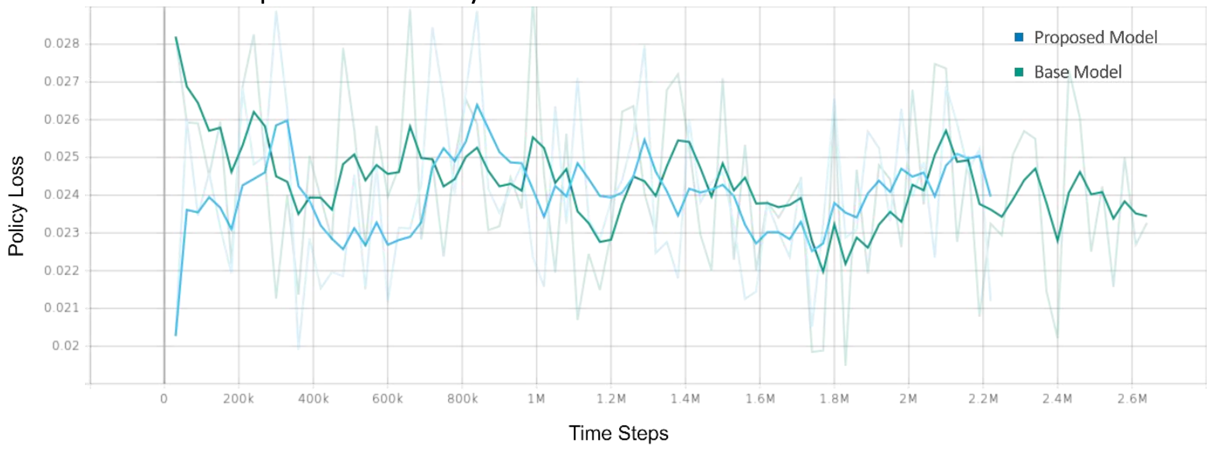

Fig. 7. Policy loss.

The value estimate is about predicting the future reward of each state. A high-value estimate means the state is good and it will lead to a maximum reward of the state and a low-value estimate means the state is bad and it will lead to minimum reward. Fig. 8 indicates the proposed model better predicts the value of the states over the base model.

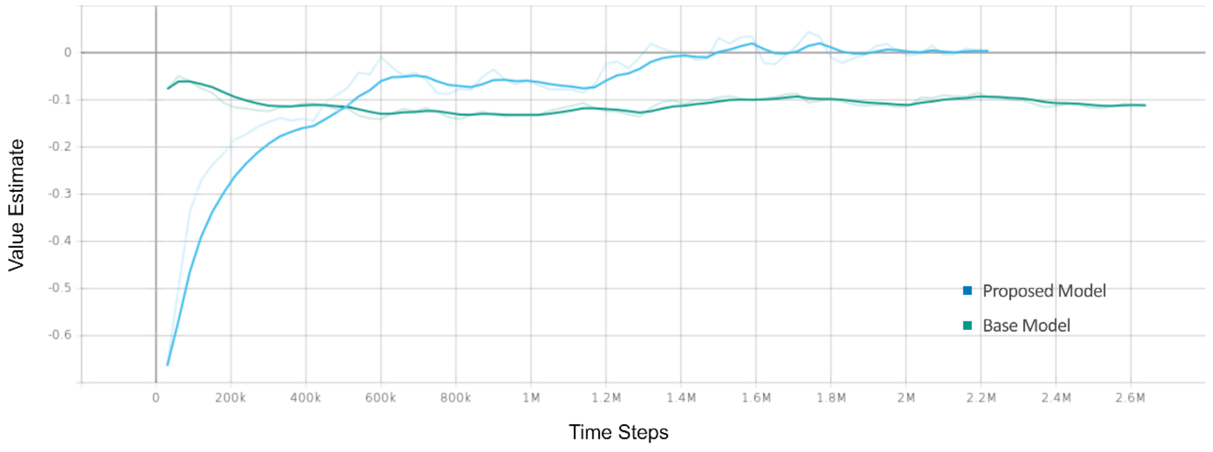

Fig. 8. Value estimate.

Value loss is the difference between the actual value of the state and the value estimated by the agent of the state. A high value loss means the agent value estimation of states is mostly wrong and a low-value loss means the agent value estimation of states is often correct. Fig. 9 shows proposed model has low value loss compared to the base model.

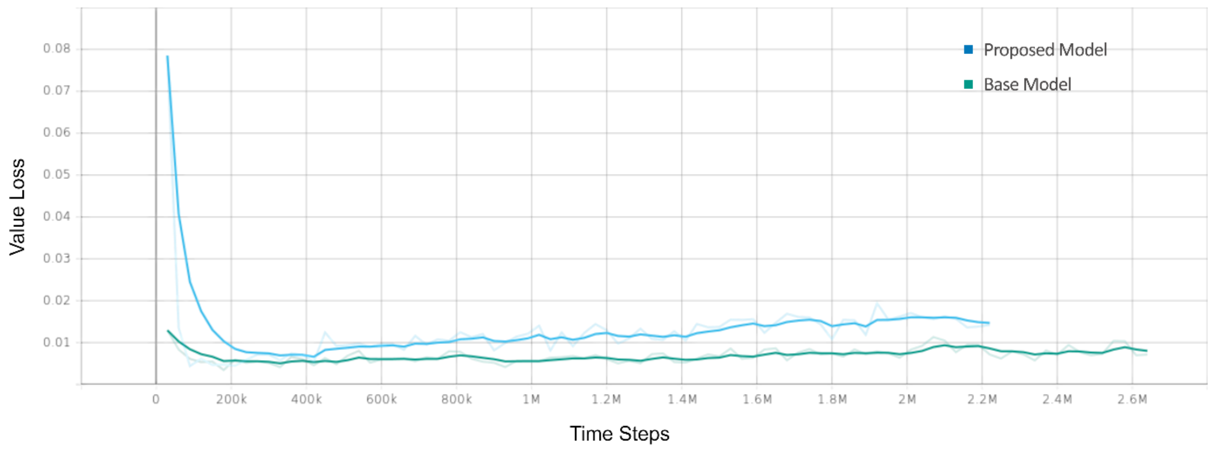

Fig. 9. Value loss.

Extrinsic rewards are the feedback which receives the agent from the environment. It shows how the agent performs its actions according to the predefined policy. This feedback helps the agent to maximize the reward received. Fig. 10 shows the proposed model performs well compared to the base model, the proposed model achieves a maximum extrinsic reward of 0.031 and whereas the base model achieves -0.27.

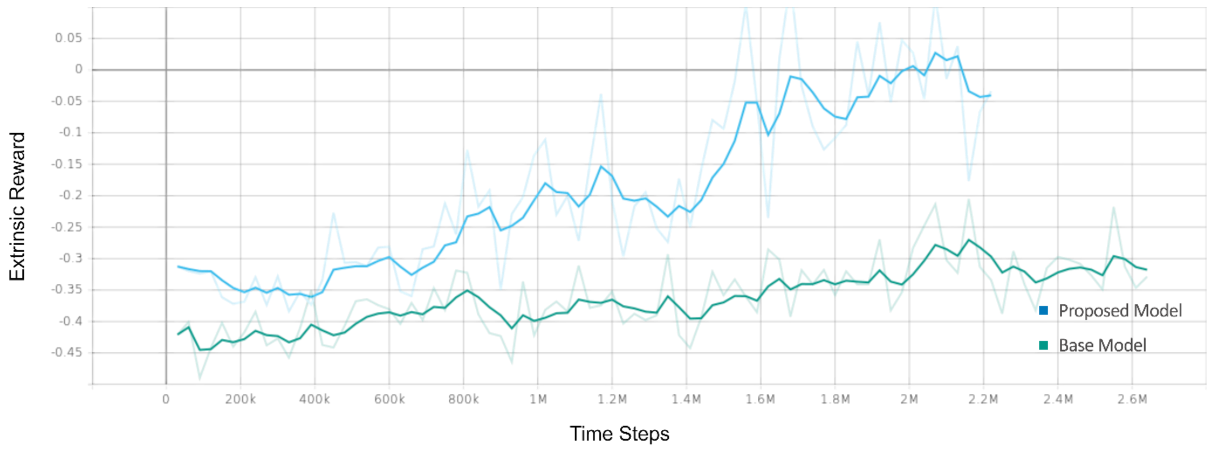

Fig. 10. Extrinsic Reward.

## 5. CONCLUSION

In Multi-Agent Systems (MAS), the role of agents is important. However, in many real-world applications, agents have different architectures, sizes, and tasks. Past research has focused on the cooperation of homogeneous agents and proposed various deep reinforcement learning techniques, which are inapplicable to heterogeneous agents due to their variation in size, architecture, and other factors. In this paper, we propose a novel approach using the PPO algorithm, along with visual and vector observations, for the collaboration of heterogeneous agents. The learning environment is created in Unity 3D. The results indicate that the proposed model outperforms the base model in terms of mean reward, extrinsic reward, and episode length. In terms of policy loss, both models perform similarly.

**Author Contributions**

All authors contributed equally.

**Funding**

This research received no external funding

## Data Availability Statement

No data available.

## Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References

[1] [1] R. S. Sutton and A. G. Barto, Reinforcement learning: An introduction. MIT press, 2018.

[2] M. M. LaMar, “Markov decision process measurement model,” Psychometrika, vol. 83, no. 1, pp. 67–88, 2018.

[3] Y. Jaafra, A. Deruyver, J. L. Laurent, and M. S. Naceur, “Context-aware autonomous driving using meta-reinforcement learning,” in 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, pp. 450–455, 2019.

[4] H. Shi, et al., “Reinforcement learning with evolutionary trajectory generator: A general approach for quadrupedal locomotion,” IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 3085–3092, 2022.

[5] C. Spatharis and K. Blekas,” Multiagent reinforcement learning for autonomous driving in traffic zones with unsignalized intersections,” Journal of Intelligent Transportation Systems, vol. 28, no. 1, pp. 103- 119, 2024.

[6] M. Khani, et al.,” Resource allocation in 5G cloud-RAN using deep reinforcement learning algorithms: A review,” Transactions on Emerging Telecommunications Technologies, vol. 35, no. 1, pp. e4929, 2024.

[7] R. Liu, F. Nageotte, P. Zanne, M. de Mathelin, and B. Dresp-Langley,” Deep reinforcement learning for the control of robotic manipulation: a focussed mini-review,” Robotics, vol. 10, no. 1, p. 22, 2021.

[8] D. Silver, et al., “Mastering the game of go with deep neural networks and tree search,” nature, vol. 529, no. 7587, pp. 484–489, 2016.

[9] V. Mnih, et al., “Human-level control through deep reinforcement learning,” nature, vol. 518, no. 7540, pp. 529–533, 2015.

[10] A.-F. Jimenez, P.-F. Cardenas, A. Canales, F. Jimenez, and A. Portacio, “A survey on intelligent agents and multi-agents for irrigation scheduling,” Computers and Electronics in Agriculture, vol. 176, p. 105474, 2020.

[11] N. Yang, et al.,” Beyond the Edge: An Advanced Exploration of Reinforcement Learning for Mobile Edge Computing, its applications, and Future Research Trajectories,” IEEE Communications Surveys & Tutorials, 2024.

[12] G. Papoudakis, F. Christianos, A. Rahman, and S. V. Albrecht,” Comparative evaluation of multi-agent deep reinforcement learning algorithms,” Autonomous Agents and MultiAgent Systems (AAMAS), pp. 1531- 1539, 2020.

[13] S. Gronauer and K. Diepold, “Multi-agent deep reinforcement learning: a survey,” Artificial Intelligence Review, pp. 1–49, 2022.

[14] D. Wang and H. Deng, “Multirobot coordination with deep reinforcement learning in complex environments,” Expert Systems with Applications, vol. 180, p. 115128, 2021.

[15] E. A. O. Diallo, A. Sugiyama, and T. Sugawara, “Coordinated behaviour of cooperative agents using deep reinforcement learning,” Neurocomputing, vol. 396, pp. 230–240, 2020.

[16] P. Stone and M. Veloso, “Multiagent systems: A survey from a machine learning perspective,” Autonomous Robots, vol. 8, pp. 345–383, 2000.

[17] B. R. Kiran, et al., “Deep reinforcement learning for autonomous driving: A survey,” IEEE Transactions on Intelligent Transportation Systems, vol. 23, no. 6, pp. 4909–4926, 2021.

[18] P. G. Morato, C. P. Andriotis, K. G. Papakonstantinou, and P. Rigo, “Inference and dynamic decision-making for deteriorating systems with probabilistic dependencies through bayesian networks and deep reinforcement learning,” Reliability Engineering & System Safety, vol. 235, p. 109144, 2023.

[19] W. Chen, X. Qiu, T. Cai, H.-N. Dai, Z. Zheng, and Y. Zhang, “Deep reinforcement learning for internet of things: A comprehensive survey,” IEEE Communications Surveys & Tutorials, vol. 23, no. 3, pp. 1659–1692, 2021.

[20] C. Zhu, M. Dastani, and S. Wang,” A survey of multi-agent deep reinforcement learning with communication,” Autonomous Agents and Multi-Agent Systems, vol. 38, no. 1, p. 4, 2024.

[21] A. Malysheva, D. Kudenko, and A. Shpilman, “Magnet: Multi-agent graph network for deep multi-agent reinforcement learning,” XVI International Symposium” Problems of Redundancy in Information and Control Systems” (REDUNDANCY). IEEE, pp. 171–176, 2019.

[22] G. Arcieri, C. Hoelzl, O. Schwery, D. Straub, K. G. Papakonstantinou, and E. Chatzi, “Bridging pomdps and bayesian decision making for robust maintenance planning under model uncertainty: An application to railway systems,” Reliability Engineering & System Safety, vol. 239, p. 109496, 2023.

[23] Z. Ye, Z. Cai, H. Yang, S. Si, and F. Zhou, “Joint optimization of maintenance and quality inspection for manufacturing networks based on deep reinforcement learning,” Reliability Engineering & System Safety, vol. 236, p. 109290, 2023.

[24] P. Trivedi and N. Hemachandra,” Multi-agent natural actor-critic reinforcement learning algorithms,” Dynamic Games and Applications, vol. 13, no. 1, pp. 25-55, 2023.

[25] Y. J. Park, Y. J. Lee, and S. B. Kim, “Cooperative multi-agent reinforcement learning with approximate model learning,” IEEE Access, vol. 8, pp. 125 389–125 400, 2020.

[26] F. Gul, et al.,” A centralized strategy for multi-agent exploration,” IEEE Access, vol. 10, pp. 126871-126884, 2022.

[27] S. H. Semnani, H. Liu, M. Everett, A. De Ruiter, and J. P. How, “Multiagent motion planning for dense and dynamic environments via deep reinforcement learning,” IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 3221–3226, 2020.