

DeepGen: A Deep Reinforcement Learning and Genetic Algorithm-Based Approach for Coverage in Unknown Environment

Nirali Sanghvi^a, Rajdeep Niyogi^b and Ribhu Mondal^c

Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, Roorkee 247667, India

Keywords: Multi-Agent Deep Reinforcement Learning, Genetic Algorithm, Coverage in Unknown Environment.

Abstract: In this paper, a novel approach to optimize waypoint placement and coverage in multi-agent systems in unknown environments using a combined Genetic Algorithm and Deep Reinforcement Learning has been proposed. Effective exploration and coverage are essential in various fields, such as surveillance, environmental monitoring, and precision agriculture, where agents must cover large and often unknown environments efficiently. The proposed method uses a Genetic Algorithm to identify optimal waypoint configurations that maximize coverage while minimizing overlap among waypoints, after which a deep reinforcement learning policy refines the agents' coverage policy to adaptively navigate and explore new areas. Simulation results demonstrate that this GA-DDQN approach significantly improves both the effectiveness of coverage and computational efficiency compared to traditional single-strategy methods. This combined framework offers a robust solution for real-world applications requiring optimized, adaptive multi-agent exploration and coverage.

1 INTRODUCTION


The coverage problem, which aims to ensure an agent visits all feasible points in an environment, is fundamental across diverse, real-world applications. These applications span critical domains such as search and rescue, space exploration, military operations, and inspection robotics as well as more routine applications like autonomous cleaning, agricultural field management, and industrial automation. The objective in these varied settings is to maximize coverage efficiently, capturing a comprehensive representation of the environment through an optimal sequence of movements, or "coverage path," to fulfill the task at hand. Coverage tasks may occur in environments that are either fully known or unknown prior to deployment, depending on the specific requirements.


In known environments, predefined factors like obstacle placement and boundaries enable offline-optimized, pre-planned coverage strategies. For instance, (Mannadiar and Rekleitis, 2010) uses Boustrophedon cellular decomposition, while (Karapetyan et al., 2017) proposes heuristic methods: one extends single-agent exact cellular decomposition, and


the other divides the environment among agents using a greedy approach. However, in partially known or unpredictable environments, such as post-disaster scenarios, agents must devise online coverage paths, dynamically adjusting movements to maximize coverage without prior information.

The use of multiple agents, rather than a single agent, has proven advantageous for improving efficiency in time-sensitive or large-scale coverage tasks. Single-agent approaches, including cellular decomposition, grid-based coverage, and graph-based coverage (Galceran and Carreras, 2013), can effectively cover small areas but lack scalability for expansive terrains or urgent situations such as search and rescue operations. Multi-agent systems, on the other hand, can enhance efficiency by reducing task completion times and improving system robustness. However, multi-agent coverage introduces additional complexity, including the risk of overlap (agents covering the same areas redundantly) and dynamic obstacles created by other moving agents. The multi-agent coverage problem can be mapped to a set of multiple Traveling Salesperson Problems, making it an NP-hard challenge (Rekleitis et al., 2008).

More recently, Reinforcement Learning (RL) has emerged as a powerful framework for solving coverage problems, offering a model in which agents can autonomously learn optimal actions through interac-

^a  <https://orcid.org/0009-0003-4245-0587>

^b  <https://orcid.org/0000-0003-1664-4882>

^c  <https://orcid.org/0009-0008-6779-2694>

tions with their environment. In RL, agents receive rewards or penalties based on their actions, iteratively refining their strategies to maximize cumulative rewards (Ladosz et al., 2022). The advent of Deep Reinforcement Learning (DRL), which incorporates deep learning to handle high-dimensional state spaces, has broadened RL’s applicability, enabling agents to operate in complex environments such as agricultural fields, where they must navigate large, intricate terrains (Li, 2017). DRL can support tasks such as autonomous field exploration, pest control, crop monitoring, and resource allocation, fostering optimal coverage and adaptive, efficient resource management.

Multi-Agent Deep Reinforcement Learning (MADRL) extends DRL principles to systems of multiple agents, enabling collaborative and competitive interactions that optimize coverage tasks in distributed, decentralized environments like agricultural fields (Gronauer and Diepold, 2022). In this paper, we propose a MADRL-based approach to coverage in agricultural fields, aiming to minimize redundancy through reduced overlap and improve efficiency via an innovative reward function. This reward function encourages cooperative coverage among agents, maximizing coverage efficiency and significantly enhancing task performance. Genetic Algorithm is an optimization approach which follows the process of natural evolution. The strongest one survives same is done by genetic algorithm. Out of all the possible prospective solutions, it selects the best solution as the final output.

In this paper, we propose an approach for coverage in unknown environments using genetic algorithm and deep reinforcement learning. Extensive simulations considering various environment sizes and varying other parameters to test the efficiency of the proposed approach. The rest of this article is structured as follows. Related work is discussed in Section 2. Formalization of the proposed approach in Section 3. The proposed approach is given in Section 4. Simulation results are given in Section 5 and conclusions in Section 6.

2 RELATED WORK

Coverage problems, known to be NP-hard, have attracted significant research interest. These problems are often simplified to the Travelling Salesman Problem (TSP) or the lawn mowing problem, highlighting their computational complexity as NP-hard. Traditionally, many area coverage approaches assume that a complete map of the environment is available, enabling efficient navigation and coverage planning.

Several traditional strategies have been explored to address the coverage problem. The greedy approach, for instance, prioritizes the nearest unvisited points, offering quick coverage but often yielding suboptimal results. A pairing method divides the environment into distinct regions, assigning each to a pair of agents; while effective, this method is feasible only when the agent-to-region ratio is balanced and may be impractical for large or unknown environments. Alternatively, exhaustive coverage methods (brute-force approaches) can ensure complete coverage but are computationally intensive and impractical for large-scale deployments (Sharma and Tiwari, 2016).

Researchers have proposed various multi-robot strategies for coverage in unknown environments. Common methods include decomposition techniques, such as Voronoi partitions for dividing areas among robots (Guruprasad et al., 2012), and sweep-based coverage using systematic sweeping paths (Sanghvi et al., 2024). A distributed approach assigns robots start and goal positions, enabling autonomous navigation based on their location (Sanghvi and Niyogi, 2024). However, suboptimal direction choices can hinder complete coverage. The spanning tree approach for multi-robot coverage was introduced in (Agmon et al., 2006). A bio-inspired method with ant-like robots marking paths for others was proposed for unknown environments in (Senthilkumar and Bharadwaj, 2012). Multi-spanning tree coverage methods, including simultaneous and extended variations, are discussed in (Chibin et al., 2008), while (Li et al., 2022) details a credit-based approach for robots with varying speeds. In (Nair and Guruprasad, 2020), a cooperative method combines Voronoi partitions with frontier-based exploration for simultaneous exploration and coverage.

Reinforcement learning provides an effective solution to address the limitations of traditional coverage methods. When combined with deep learning, reinforcement learning becomes a powerful tool for coverage tasks in large, complex environments. In (Wang et al., 2023), the authors propose a coverage path planning approach using DQN, specifically adapted for a targeted task. In (Piardi et al., 2019) employs Q-learning to determine optimal coverage paths while aiming to avoid overlapping areas, though it assumes known obstacle positions. In contrast, our proposed approach operates without prior knowledge of the environment, barriers, or the positions of other agents, allowing for more autonomous exploration. In (Din et al., 2022) presents a DDQN-based method for multi-agent coverage, allowing certain areas to be revisited. However, in many coverage tasks, revisiting the same location may be inefficient and resource-

intensive. Our approach utilizes deep reinforcement learning within a discrete action space, providing a practical and efficient solution for complex scenarios.

In (Such et al., 2017), the authors have proposed an approach for using genetic algorithm for training of neural networks. There the authors present that using genetic algorithm helps in better and faster training the neural network and this approach can work well with various deep reinforcement learning approaches. In (Sehgal et al., 2019), the authors have used genetic algorithm for parameter optimization of deep reinforcement learning. In this paper, we combine the genetic algorithm approach with deep reinforcement learning for the coverage problem of an unknown environment.

3 FORMALIZATION

Definition 1. Waypoints Waypoints are the intermediate targets that guide the movement of the agents to increase the overall coverage.

Definition 2. Coverage (C_g) Let C^i be the area covered by agent i . Coverage C_g is the union of the areas covered by each agent, i.e., $C_g = \bigcup_{i=1}^n C^i$

Definition 3. Coverage Percentage ($C_{\%}$) Coverage Percentage ($C_{\%}$) is the ratio of Coverage (C_g) to the total area (A) that needs to be covered.

$$C_{\%} = \frac{C_g}{A} \times 100$$

3.1 Problem Definition

Let I denote the set of agents $I = \{1, \dots, n\}$, where n denotes the total number of agents in the given environment. Let \mathcal{W} be the set of waypoints $\mathcal{W} = \{w_1, \dots, w_k\}$, where k denotes the total number of waypoints placed. Let O denote the set of obstacles that are placed randomly in the environment. The agents have no knowledge about the placement of the obstacles in the environment. Also, the agents do not have any information about the position of other agents or the placement of the waypoints. The main goal is to attain better coverage of the environment.

3.2 Problem Formalization

We propose an approach using a genetic algorithm and Deep Reinforcement Learning (DRL) to address the coverage problem. In this proposed approach, the waypoints are placed in the environment using the genetic algorithm (GA). The main goal is to maximize the coverage and get better coverage with reduced

overlap for which DRL is used. In this, the agent I interacts with the environment using its individual policy based on its local observations. The goal is to maximize the rewards and thus attain better coverage and reduced overlap. At a given time, the agents collectively take a joint action $\mathbf{a}_t = (a_t^1, \dots, a_t^n) \in \mathcal{A}$, where \mathcal{A} is the action space composed of individual action spaces \mathcal{A}_i for agent i . Then, the agents transition to a new state $s' \in \mathcal{S}$ with a probability distribution of $P(s'|s, a)$. Upon transitioning to the new state, the agent receives a reward r_t .

4 PROPOSED APPROACH

4.1 Genetic Algorithm

Figure 1 gives the overview the proposed approach. When multiple agents need to explore and cover an unknown environment, a Genetic Algorithm (GA) can optimize the placement of waypoints to maximize coverage while minimizing overlap. Because the environment layout is initially unknown, the GA iteratively refines waypoint configurations, aiming for solutions that allow agents to efficiently cover the area. Each configuration (or individual) in the GA population represents a potential waypoint arrangement that directs agents' paths to achieve effective coverage. The algorithm includes five key stages: generating an initial population, evaluating fitness, selecting high-performing configurations, applying crossover to create new configurations, and using mutation to maintain diversity and explore the solution space. Below are the important functions used.

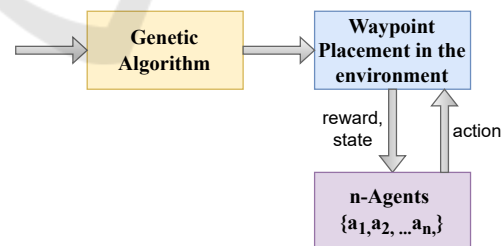


Figure 1: Proposed Approach.

4.1.1 Initial Population

The initial population represents a varied set of candidate waypoint configurations, where each configuration is a different arrangement of waypoints for agent deployment. Given that the environment's layout and potential obstacles are not known beforehand, these configurations are generated randomly to provide a wide range of possible solutions. Each configuration

suggests different points where agents might begin or move toward, encouraging broad exploration. This diversity is essential in unknown environments as it offers a range of strategies that the algorithm can evaluate and refine.

4.1.2 Fitness Function

The fitness function is a measure that evaluates each individual's quality or effectiveness in solving the problem. It assigns a fitness score to each individual based on how well they fulfill the objectives. The fitness function is crucial because it guides the selection process, helping the GA focus on individuals that show the most promise for improvement over generations. The fitness function evaluates each waypoint configuration based on its effectiveness in maximizing coverage and minimizing overlap among agents. In this coverage scenario, fitness scores are calculated by balancing two primary metrics: Waypoint Coverage (W_c) and Waypoint overlap (W_o). Waypoint Coverage (W_c) measures how well the waypoints are placed in the environment. It checks on the efficiency of the waypoint placement. Waypoint overlap (W_o) measures if the waypoints are placed in a very close proximity or if they are placed at the same locations.

The fitness function for each configuration is calculated as:

$$f(W) = \alpha \cdot W_c - \beta \cdot W_o \quad (1)$$

where α and β are constants that weigh the importance of coverage and overlap respectively. A high fitness score indicates that the waypoint configuration effectively maximizes coverage while minimizing overlap.

4.1.3 Selection

Selection is a process by which individuals with higher fitness scores are chosen as parents for the next generation, as they are more likely to produce offspring with advantageous traits. Selection aims to prioritize individuals that are closer to the optimal solution, increasing the probability that beneficial traits will be passed down.

4.1.4 Crossover

Crossover is a genetic operation applied to the selected parents to produce the next generation. Crossover combines segments of genetic information from two parents to generate offspring, which inherit characteristics from both. This operation enables exploration of the solution space by creating new individuals with potentially improved traits.

4.1.5 Mutation

Mutation introduces random alterations to the genetic makeup of offspring, typically with a small probability. Mutation prevents premature convergence to local optima by introducing genetic diversity, which helps the population explore a broader solution space. This random change might involve modifying a waypoint position or altering a path sequence. By maintaining a low mutation rate, the algorithm balances diversity with stability, ensuring that beneficial traits from previous generations are retained while still exploring new solutions.

4.2 Deep Reinforcement Learning Based Coverage

State Space: The state space for the coverage problem is the set of all states in the environment. State space in the grid of size $M \times N$ is all the cells present in the grid, which may be either covered, uncovered, or have obstacles.

Action Space: For every agent, there are four discrete actions *up*, *down*, *right*, *left*.

Reward: The reward in DDQN represents the environment's response to an agent's action and significantly influences learning efficiency and behavior. A well-designed reward function promotes efficient coverage, exploration, and collaboration, particularly in multi-agent systems. Dense rewards provide immediate feedback, accelerating learning and convergence. Adjusting the reward structure balances trade-offs such as coverage efficiency, energy use, or collision avoidance, encouraging faster convergence and mitigating overfitting or unintended behaviors. In multi-agent settings, rewards emphasizing team performance enhance coordination, while penalizing redundancy ensures effective collaboration.

The reward function depends on the number of waypoints and the total area covered by the agents. They get more reward on exploring unknown area which indeed improvises the coverage. We model the reward function. The reward function R_t at each time step t is defined as:

$$R_t = \alpha \cdot W_c + \beta \cdot C_g$$

where α and β are weighing factors that balance the importance of total coverage and exploration.

In this function, the term $\alpha \cdot W_c$ rewards agents based on overall coverage, motivating them to maximize the total area covered. The term $\beta \cdot C_g$ adds an incentive for exploring previously unexplored regions, thus encouraging agents to seek new areas rather than revisiting known ones. By tuning α and β , the reward

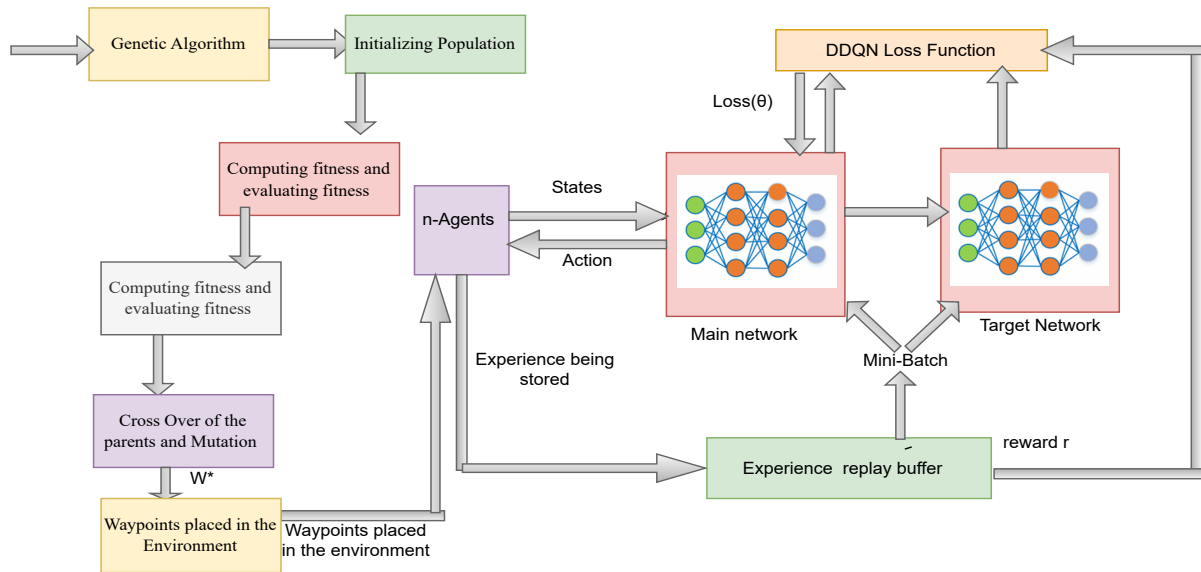


Figure 2: Working of the proposed approach.

function can prioritize between broad coverage and the exploration of unknown areas, enhancing efficient exploration. This cycle of selection, crossover, and mutation repeats until the specified maximum number of generations, G_{max} , is reached. At this point, the best-performing waypoint configuration W^* based on the highest fitness score, is selected as the final output.

4.3 Algorithm

The GA-DDQN algorithm optimizes waypoint placement and coverage in a grid environment by combining Genetic Algorithm (GA) for global optimization and Double Deep Q-Network (DDQN) for policy refinement. In Stage 1, GA determines the optimal waypoint configuration W^* by iteratively generating and evaluating random populations of configurations. Fittest individuals undergo selection, crossover, and mutation, and the process continues for G_{max} generations to yield W^* . In Stage 2, DDQN refines the coverage policy using W^* . Initialized with parameters $(E, S_{max}, \alpha, \gamma, \epsilon)$, an agent interacts with the environment, learning from experiences stored in a replay buffer. Training continues until the coverage target or step limit is reached, leveraging DDQN's ability to handle large state spaces efficiently. The result is an optimized waypoint configuration W^* and a refined coverage policy maximizing grid efficiency.

Figure 2 illustrates the workflow of the proposed GA-DDQN (Genetic Algorithm-Double Deep Q-Network) framework for optimizing waypoint placement and coverage. The process begins with the Genetic Algorithm (GA) component, where an initial population of waypoint configurations is gener-

ated. Each configuration undergoes fitness evaluation, where coverage effectiveness is assessed. Using selection, crossover, and mutation, the GA evolves the waypoint configurations, producing an optimized set, denoted as W^* , which is then placed in the environment.

Once waypoints are set, multiple agents interact with the environment, gathering experiences by navigating through the waypoints and avoiding obstacles. Each agent's interaction yields states and actions, which are processed by the DDQN module. The DDQN consists of a Main Network and a Target Network. The agents' actions and resulting states, along with the obtained rewards r , are stored in an experience replay buffer. Periodically, mini-batches from this buffer are used to train the DDQN, minimizing the loss function $Loss(\theta)$, which updates the network parameters and refines the policy for optimal navigation and coverage. This proposed approach using GA for waypoint optimization and the DDQN for adaptive policy learning allows for effective navigation, maximizing coverage efficiency while reducing redundant overlaps.

5 EXPERIMENTS AND RESULTS

In this section, we present the simulation results of our proposed approach. Figure 3 shows a 10x10 grid environment used for simulation, featuring various elements relevant to a coverage task. Black cells represent obstacles that add complexity by restricting movement, simulating real-world barriers. Blue dots denote waypoints, which are key locations that agents

Algorithm 1: GA-DDQN for Optimizing Waypoint Placement and Coverage.

Input : Grid environment G , number of agents N , obstacles O , waypoints W , GA parameters (P, μ, G_{max}) , DDQN parameters $(E, S_{max}, \alpha, \gamma, \epsilon, \epsilon_{decay}, \epsilon_{min})$

Output: Optimal waypoint configuration and DDQN-based coverage policy

```

1 for  $i = 1$  to  $P$  do
2   Generate a random set of waypoints  $W_i$ ;
3   Add  $W_i$  to initial population;
4 for  $j = 1$  to  $G_{max}$  do
5   for each individual  $W_i$  in population do
6     Compute fitness for  $W_i$  using
7     Equation( 1)
8   Select fittest individuals;
9   for each individual in new population do
10    Select two parents  $P_1$  and  $P_2$  based on
11    fitness;
12    Generate child by
13    CROSSOVER( $P_1, P_2$ );
14    Mutate child with probability  $\mu$ ;
15    Add child to new population;
16  Update population with new population;
17 Select best waypoint configuration  $W^*$  from
18  population;
19 for  $e = 1$  to  $E$  do
20  Reset environment with waypoints  $W^*$ ;
21  for  $t \leftarrow 1$  to  $S_{max}$  do
22    Observe state  $s_t$ ;
23    Select action  $a_t$  based on DDQN
24    policy;
25    Execute  $a_t$ , observe next state  $s_{t+1}$ 
26    and reward  $r_t$ ;
27    Store transition  $(s_t, a_t, r_t, s_{t+1})$  in
28    replay buffer;
29    Perform DDQN update using
30    mini-batch from replay buffer;
31    if coverage target reached or done
32    then
33      Break;
    
```

aim to cover. The green cell in the bottom-left corner likely marks the starting position, while the red cell in the top-right corner serves as the target or endpoint. This setup challenges agents to navigate around obstacles and maximize waypoint coverage, providing a basis for assessing the effectiveness of different coverage strategies. During the simulation, other agents

may also serve as dynamic obstacles, further complicating navigation and coverage. The coverage task was executed with varying numbers of agents using the proposed approach. All simulations were conducted on a core-i7 processor with 32 GB RAM.

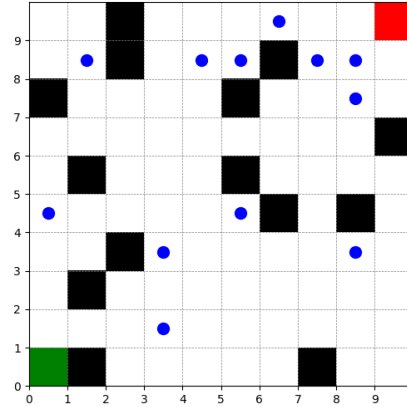


Figure 3: Waypoint Placement using Random Approach.

In figure 3 waypoints (blue dots) are placed randomly across the grid without any optimization. The grid contains black cells representing obstacles that agents must avoid. The green cell in the bottom-left corner likely indicates the starting point for agents, while the red cell in the top-right corner serves as the target or endpoint. This random placement may lead to suboptimal coverage and inefficient movement paths, as waypoints may be clustered or unevenly distributed, potentially requiring agents to revisit certain areas or navigate inefficiently around obstacles. The coverage obtained in this is 57.8%.

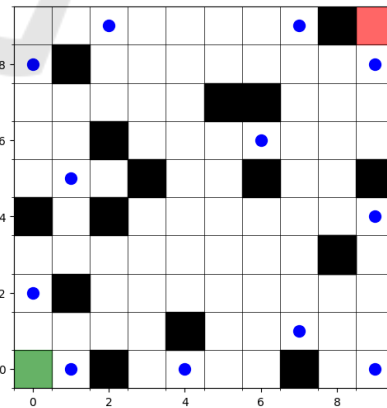


Figure 4: Waypoint Placement using GA.

In figure 4, Genetic Algorithm (GA) is used to optimize the placement of waypoints, resulting in a more structured distribution across the grid followed by deep reinforcement learning for coverage. By op-

timizing waypoint positions, the GA approach aims to improve coverage and minimize overlap, allowing agents to navigate more efficiently and avoid unnecessary detours. This structured placement supports better exploration and coverage, as agents can follow an optimized path that maximizes area coverage while avoiding obstacles, attaining a coverage of 87.8%.

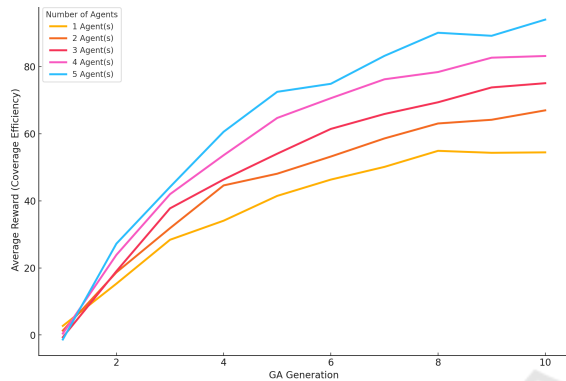


Figure 5: Reward obtained using different number of agents.

Figure 5 shows the increase in average reward (coverage efficiency) across GA generations for different agent counts (1 to 5). Each line represents a specific number of agents, with more agents achieving higher rewards. The trend indicates that, as GA generations progress, coverage efficiency improves consistently, with the 5-agent configuration reaching the highest rewards. This demonstrates the effectiveness of the GA-DDQN hybrid approach in optimizing coverage through more agents and iterative learning.

Table 1: Area Coverage (%) vs. Number of Agents.

(n)	GA Coverage (%)	DDQN Coverage (%)	GA+DDQN Hybrid Coverage (%)
1	63.1	77.5	86.3
2	62.7	72.2	89.4
3	69.6	74.8	88.6
4	72.5	81.5	92.1
5	71.3	83.3	94.5

Table 1 compares the area coverage percentages achieved by Genetic Algorithm (GA), Double Deep Q-Network (DDQN), and the hybrid GA+DDQN method across varying agent counts (n). With one agent, GA achieves 63.1% coverage, DDQN performs better at 77.5%, and the hybrid method outperforms both with 86.3%. With two agents, GA coverage decreases slightly to 62.7%, DDQN remains stable at 72.2%, while the hybrid approach achieves a significant

increase to 89.4%. For three agents, GA improves to 69.6%, DDQN rises to 74.8%, and the hybrid method continues to lead with 88.6% coverage.

With four agents, GA and DDQN achieve 72.5% and 81.5% coverage, respectively, while the hybrid approach reaches 92.1%. At five agents, GA achieves 71.3%, DDQN improves to 83.3%, and the hybrid approach achieves its highest coverage of 94.5%, fully utilizing the increased agent count for optimized coverage. Table 1 demonstrates the hybrid method’s superior performance by combining GA’s global waypoint optimization with DDQN’s adaptive learning, significantly enhancing coverage in multi-agent environments.

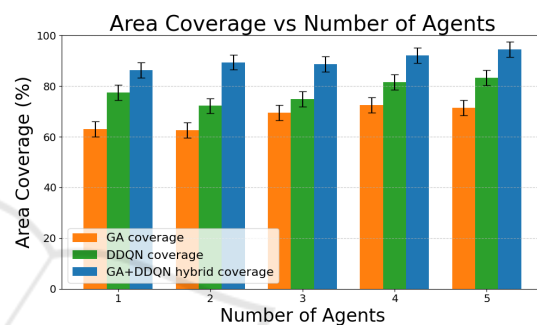


Figure 6: Coverage obtained by DQN and our proposed DDQN in the same environment.

Figure 6 shows the relationship between the number of agents and the area coverage percentage achieved by Genetic Algorithm (GA), Double Deep Q-Network (DDQN), and the hybrid GA+DDQN approach. The x-axis represents the number of agents (1 to 5), and the y-axis shows the percentage of area covered. As depicted, coverage increases with the number of agents across all methods. GA consistently achieves the lowest coverage, highlighting its limitations in independently covering the area effectively. DDQN performs better than GA, achieving higher coverage as the number of agents increases, but it still lags behind the hybrid method. The hybrid GA+DDQN approach combines GA’s global waypoint optimization with DDQN’s adaptive exploration, achieving the highest coverage percentages for all agent counts.

6 CONCLUSIONS

In this paper, a hybrid GA-DDQN approach for optimizing waypoint placement and coverage in a grid-based environment has been developed. Using Genetic Algorithm for waypoint optimization with Double Deep Q-Network for adaptive policy learning,

the approach achieves efficient navigation and enhanced coverage, outperforming traditional single-method approaches. Our results show that coverage ranges from approximately 86.3% with one agent to 94.5% with five agents, demonstrating significant improvements with increasing the number of agents.

As a part of future work, we aim to extend the proposed algorithm to handle environments where the obstacles are moving.

ACKNOWLEDGMENT

The second author was in part supported by a research grant from Google.

REFERENCES

- Agmon, N., Hazon, N., and Kaminka, G. A. (2006). Constructing spanning trees for efficient multi-robot coverage. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1698–1703. IEEE.
- Chibin, Z., Xingsong, W., and Yong, D. (2008). Complete coverage path planning based on ant colony algorithm. In *2008 15th International Conference on Mechatronics and Machine Vision in Practice*, pages 357–361. IEEE.
- Din, A., Ismail, M. Y., Shah, B., Babar, M., Ali, F., and Baig, S. U. (2022). A deep reinforcement learning-based multi-agent area coverage control for smart agriculture. *Computers and Electrical Engineering*, 101:108089.
- Galceran, E. and Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous systems*, 61(12):1258–1276.
- Gronauer, S. and Diepold, K. (2022). Multi-agent deep reinforcement learning: a survey. *Artificial Intelligence Review*, 55(2):895–943.
- Guruprasad, K., Wilson, Z., and Dasgupta, P. (2012). Complete coverage of an initially unknown environment by multiple robots using voronoi partition. In *International Conference on Advances in Control and Optimization in Dynamical Systems*.
- Karapetyan, N., Benson, K., McKinney, C., Taslakian, P., and Rekleitis, I. (2017). Efficient multi-robot coverage of a known environment. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1846–1852. IEEE.
- Ladosz, P., Weng, L., Kim, M., and Oh, H. (2022). Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22.
- Li, L., Shi, D., Jin, S., Kang, Y., Xue, C., Zhou, X., Liu, H., and Yu, X. (2022). Complete coverage problem of multiple robots with different velocities. *International Journal of Advanced Robotic Systems*, 19(2):17298806221091685.
- Li, Y. (2017). Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*.
- Mannadiar, R. and Rekleitis, I. (2010). Optimal coverage of a known arbitrary environment. In *2010 IEEE International conference on robotics and automation*, pages 5525–5530. IEEE.
- Nair, V. G. and Guruprasad, K. (2020). Mr-simexcoverage: Multi-robot simultaneous exploration and coverage. *Computers & Electrical Engineering*, 85:106680.
- Piardi, L., Lima, J., Pereira, A. I., and Costa, P. (2019). Coverage path planning optimization based on q-learning algorithm. In *Aip conference proceedings*, volume 2116. AIP Publishing.
- Rekleitis, I., New, A. P., Rankin, E. S., and Choset, H. (2008). Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Annals of Mathematics and Artificial Intelligence*, 52:109–142.
- Sanghvi, N. and Niyogi, R. (2024). Distributed coverage algorithm using multiple robots in an unknown environment.
- Sanghvi, N., Niyogi, R., and Milani, A. (2024). Sweeping-based multi-robot exploration in an unknown environment using webots. In *ICAART (1)*, pages 248–255.
- Sehgal, A., La, H., Louis, S., and Nguyen, H. (2019). Deep reinforcement learning using genetic algorithm for parameter optimization. In *2019 Third IEEE International Conference on Robotic Computing (IRC)*, pages 596–601. IEEE.
- Senthilkumar, K. and Bharadwaj, K. K. (2012). Multi-robot exploration and terrain coverage in an unknown environment. *Robotics and Autonomous Systems*, 60(1):123–132.
- Sharma, S. and Tiwari, R. (2016). A survey on multi robots area exploration techniques and algorithms. In *2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT)*, pages 151–158. IEEE.
- Such, F. P., Madhavan, V., Conti, E., Lehman, J., Stanley, K. O., and Clune, J. (2017). Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. *arXiv preprint arXiv:1712.06567*.
- Wang, Y., He, Z., Cao, D., Ma, L., Li, K., Jia, L., and Cui, Y. (2023). Coverage path planning for kiwifruit picking robots based on deep reinforcement learning. *Computers and Electronics in Agriculture*, 205:107593.