MULTI-UAV PATH PLANNING FOR JOINT COVERAGE AND CONNECTIVITY USING REINFORCEMENT LEARNING

İSLAM GÜVEN

ÖZYEĞİN UNIVERSITY

JANUARY, 2025

MULTI-UAV PATH PLANNING FOR JOINT COVERAGE AND CONNECTIVITY USING REINFORCEMENT LEARNING

by

İslam Güven

Thesis
Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

in
Electrical and Electronics Engineering

Advisor: Assoc. Prof. Evşen Yanmaz Adam

Graduate School of Science and Engineering
Özyeğin University
İstanbul

January, 2025

MULTI-UAV PATH PLANNING FOR JOINT COVERAGE AND CONNECTIVITY USING REINFORCEMENT LEARNING

Approved by:

Assoc. Prof. Evşen Yanmaz Adam, Advisor
Department of Electrical and Electronics Engineering
Özyeğin University

Assoc. Prof. Cenk Demiroğlu
Department of Electrical and Electronics Engineering
Özyeğin University

Prof. Dr. Hazım Kemal Ekenel
Department of Computer Engineering
Istanbul Technical University

Approval Date: December 24, 2024

To my advisor and mentors, whose guidance, patience, and wisdom have been invaluable in shaping both this thesis and my scientific development.

DECLARATION OF ORIGINALITY

I hereby declare that I am the sole author of this thesis and that this is the true copy of my thesis, including the final revisions, approved by my thesis committee. All data and information have been obtained, produced, and presented in accordance with the rules of research ethics and principles of academic honesty. As required by these rules, to the best of my knowledge I have acknowledged ideas, thoughts, and any copyrighted material in accordance with the standard referencing rules. I certify that any part of this thesis has not been submitted for a degree or diploma in another educational institution.

İslam Güven

ABSTRACT

This thesis presents a multi-objective optimization (MOO) framework in multi-Unmanned Aerial Vehicle (UAV) path planning, transitioning from classical optimization methods to advanced reinforcement learning techniques. We address two main challenges in multi-UAV systems: the need for real-time adaptability and the balance between competing mission objectives. First, we develop a new multi-objective optimization framework that simultaneously considers coverage time and network connectivity. This framework provides a diverse set of Pareto-optimal solutions, highlighting meaningful trade-offs between different mission goals. Building on these findings, we introduce a dynamic reinforcement learning system using Deep Q-Networks that allows UAVs to adjust their behavior according to changing mission requirements. This system demonstrates improvement in target detection times. Finally, we present an advanced Proximal Policy Optimization framework that effectively manages heterogeneous targets with varying information priorities. This framework includes a dynamic reward mechanism that integrates area exploration, UAV-Ground Control Station (GCS) connectivity, and target-specific requirements. Our experimental results show that the proposed framework remains computationally efficient while scaling to larger teams of up to 20 UAVs and adapting to different target configurations. The progression from static optimization through basic reinforcement learning to advanced adaptive systems not only enhances the technical capabilities of multi-UAV systems but also offers valuable implementation insights for researchers and practitioners in the field.

Keywords: Multi-UAV Systems, Path Planning, Reinforcement Learning, Multi-objective Optimization, Adaptive Control, Network Connectivity

ÖZET

Bu tezde, aralarında klasik optimizasyon ve gelişmiş pekiştirmeli öğrenme tekniklerinin de olduğu farklı çoklu İnsansız Hava Aracı (İHA) rota planlama yöntemleri önerilmektedir. Buna yönelik olarak çoklu İHA sistemlerindeki iki ana zorluğa odaklanılmaktadır: Gerçek zamanlı adaptasyon ihtiyacı ve birbiriyle çelişen görev amaçları arasında bir denge kurma gerekliliği. İlk olarak, kapsama süresi ve ağ bağlantısını eş zamanlı değerlendiren yenilikçi bir çok amaçlı optimizasyon sistemi geliştirilmiştir. Bu sistem, farklı görev hedefleri arasındaki ilişkileri matematiksel olarak anlamlandırarak, geniş bir Pareto-optimal çözüm kümesi sunmaktadır. Daha sonra, bu bulgular üzerine, İHAların değişen görev gereksinimlerine göre davranışlarını uyarlamalarına imkan tanıyan, Derin Q-Ağları tabanlı dinamik bir pekiştirmeli öğrenme sistemi önerilmiş ve bu sistemin, hedef tespit sürelerinde geleneksel yöntemlere kıyasla iyileşme sağladığı gözlemlenmiştir. Son olarak, farklı bilgi önceliklerine sahip çoklu hedefleri etkili bir şekilde takip edebilen gelişmiş bir Yaklaşık Politika Optimizasyonu sistemi önerilmiştir. Bu sistem, alan taraması oranı, İHA-Yer Kontrol İstasyonu bağlantısı ve hedefe özgü gereksinimleri bütünleştiren dinamik bir ödül mekanizması içermektedir. Performans sonuçları, önerilen bu sistemin 20 İHAya kadar takımlara ölçeklenebilip farklı hedef türlerine uyum sağlayabilirken, hesaplama süresi açısından verimliliğini koruduğunu göstermektedir. Klasik optimizasyondan başlayıp temel pekiştirmeli öğrenmeye geçerek gelişmiş adaptif sistemlere doğru tezde kaydedilen bu ilerlemenin, çoklu İHA sistemlerinin teknik kabiliyetlerini geliştirmekle kalmayıp, aynı zamanda alandaki araştırmacılar ve uygulayıcılar için de faydalı uygulama yöntemleri sağlayacağı öngörülmektedir.

Anahtar Kelimeler: Çoklu-İHA Sistemleri, Rota Planlama, Pekiştirmeli Öğrenme, Çok Amaçlı Optimizasyon, Adaptif Kontrol, Ağ Bağlantısı

ACKNOWLEDGEMENTS

Foremost, I sincerely thank my supervisor, Assoc. Prof. Evşen Yanmaz Adam, for her exceptional scientific and technical support, her thoughtful guidance, and her remarkable project management skills. Her dedication to academic excellence and support for innovative research approaches have been instrumental in shaping this project.

I would also like to acknowledge Assoc. Prof. Mehmet Parlak, whose earlier mentorship and technical insights provided a valuable foundation for this research endeavor. Lastly, I want to express my gratitude to our Dean, Prof. Hasan Fatih Uğurdağ, for his support and guidance.

This study was funded by the The Scientific and Technological Research Council of Turkey (TUBITAK) Academic Research Funding Programmes Directorate (ARDEB) 1001 Grant No 121E408.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS AND ABBREVIATIONS

A2C	Advantage Actor-Critic
ACO	Ant Colony Optimization
AI	Artificial Intelligence
ARDEB	Academic Research Funding Programmes Directorate
CPU	Central Processing Unit
CNN	Convolutional Neural Network
DDPG	Deep Deterministic Policy Gradient
DQN	Deep Q-Network
GA	Genetic Algorithm
GCS	ground control station
IoT	Internet of Things
MARL	Multi-Agent Reinforcement Learning
MOEA	Multi-Objective Evolutionary Algorithm
MOO	Multi-Objective Optimization
NSGA	Non-Dominated Sorting Genetic Algorithm
PPO	Proximal Policy Optimization
PSO	Particle Swarm Optimization
RAM	Random Access Memory
RL	Reinforcement Learning
SAR	Search and Rescue

SOO Single-Objective Optimization

TS Tabu Search

TUBITAK The Scientific and Technological Research Council of Turkey

UAV Unmanned Aerial Vehicles

VNS Variable Neighborhood Search

WS Weighted Sum

1. INTRODUCTION

The development of autonomous systems has fundamentally transformed our approach to complex real-world problems. Unmanned Aerial Vehicles (UAV) are at the forefront of this transformation - autonomous devices capable of perceiving their environment, making decisions, and acting without constant human intervention [1]. While single UAVs have proven valuable for tasks such as aerial photography and infrastructure inspection, the true potential of this technology emerges when multiple UAVs operate as a coordinated team [2, 3].

Over the years of deployment, autonomous UAV systems have evolved significantly, incorporating advanced sensing technologies alongside complex decision-making algorithms [4]. These systems can process visual information in real time, navigate challenging terrain, and maintain reliable communication links even under disaster conditions. This evolution has made UAVs particularly valuable for Search and Rescue (SAR) missions, where teams must efficiently search large areas while maintaining reliable communication networks.

1.1 Motivation

In SAR operations, mission success depends on two critical capabilities: area coverage and information relay. UAVs must explore the mission zone to locate various targets of varying priorities, while ensuring that all collected information reaches a central GCS [1]. Unlike conventional remotely piloted aircraft that operate under direct human supervision within visual line of sight, autonomous UAV teams must function beyond direct communication range. This necessitates multi-hop communication, where information flows through a network of UAVs acting as relay points [5]. This creates a fundamental conflict between exploration and connectivity where UAVs must maintain communication

links through direct or multi-hop connections to the GCS while efficiently covering the search area. This trade-off forms the core Multi-Objective Optimization (MOO) problem investigated in this thesis.

We can observe this problem in disaster situations. Consider a post-disaster scenario: A fleet of autonomous UAVs operates above devastated urban terrain, each equipped with advanced sensors including cameras for survivor detection, thermal imagers for structural assessment, and wireless communication devices. As UAVs navigate the disaster zone, they must cover the entire area quickly while maintaining an optimal network topology to relay critical information back to the GCS [6, 7]. To maximize search efficiency, the UAVs should spread widely across the disaster zone. However, if a UAV discovers a survivor but cannot relay this information back to rescue teams, the discovery becomes meaningless. The trade-off between exploration and connectivity shows the need for a generic solution framework that is both adaptable and modular in design.

This thesis systematically builds this framework with the addition of dynamic mission elements. Our investigation ranges from traditional optimization approaches to advanced Artificial Intelligence (AI)-based path planners, demonstrating how we address these challenges while maintaining practical applicability. Our work offers valuable information for robotics researchers and practitioners interested in deploying UAV teams for disaster response operations.

1.2 Problem Statement and Main Contributions

The evolution of UAV deployment in disaster response as shown in Table. 1.1 reveals several fundamental gaps that shape our research direction. Early deployments in 1995 relied on human operators maintaining constant visual contact, severely limiting operational range and complexity. The recent Turkey-Syria earthquake response in 2023 particularly highlighted how preplanned flight paths often failed when faced with

Table 1.1: *Sample UAV Deployments in Disaster Response (1995–2023).*

Year	Event Description	Operation Type
1995	Deployment of human-operated UAVs for aerial surveillance in disaster scenarios. [8]	Non-Autonomous
2005	Use of remotely piloted UAVs for damage assessment during Hurricane Katrina. [9]	Non-Autonomous
2010	Application of human-controlled drones for mapping affected areas post-Haiti earthquake. [9]	Non-Autonomous
2015	Implementation of semi-autonomous UAVs for 3D mapping to assist relief efforts in Nepal earthquake. [10]	Autonomous
2017	Deployment of autonomous UAVs for damage assessment during Hurricanes Harvey and Irma. [9]	Autonomous
2020	Use of AI-powered drones to autonomously detect hotspots during Australian bushfires. [11]	Autonomous
2023	Deployment of autonomous UAVs with AI for search and rescue missions in Turkey-Syria earthquake. [12]	Autonomous

collapsed buildings and dynamic obstacles [12]. UAVs struggled to maintain reliable data links while exploring vastly damaged areas [6], and the complexity of coordination increased dramatically with team size, severely limiting effective deployment [7].

These real-world challenges led us to explore three interconnected research topics that progressively build upon each other. First, we investigated how to effectively balance the inherent trade-off between area exploration and connectivity maintenance while keeping computational requirements manageable. Traditional approaches typically treat these as separate problems or use weighted sums to convert the problem into a Single-Objective Optimization (SOO) problem [1]. Instead, we investigate MOO problems forming a **Pareto Optimal Front**. The Pareto Optimal Front represents a set of solutions where improving one objective degrades another, providing decision-makers with a range of optimal trade-offs.

Second, we examined how UAV teams could dynamically adjust their behavior in response to environmental changes while maintaining mission objectives, addressing the limitations of static path planning systems [4]. Finally, we investigate the behavior of dynamic models trained on homogeneous target distributions evaluated under heterogeneous target distributions.

Our three main technical contributions directly address these research topics through a systematic progression from traditional optimization to advanced learning-based approaches:

- We develop a novel **MOO framework** that jointly considers both coverage and connectivity requirements. Using custom genetic operators and evolutionary algorithms, this approach provides a diverse set of solutions that capture meaningful trade-offs between competing objectives, allowing operators to choose the most suitable strategy for their specific mission requirements. The model includes a modular structure for further additions in mission details and objective functions. [13]

- We design a **dynamic Reinforcement Learning (RL) system** that enables UAVs to adapt their behavior to changing mission requirements. This approach allows us to test or model on missions aiming for detection and information of randomly placed targets. We build the Multi-Agent Reinforcement Learning (MARL) model using Deep Q-Network (DQN) combined with Convolutional Neural Network (CNN) for novel multi-layered image state processing. [14]

- We introduce an advanced **PPO framework** that scales efficiently to larger teams while handling heterogeneous targets. This approach decomposes the global reward structure into local and global components, enabling effective coordination of up to 20 UAVs.

Each of these contributions builds on the limitations identified in previous approaches [2]. Our evaluation framework assesses performance across three fundamental dimensions: Coverage time (How quickly UAVs can explore the mission area), connectivity maintenance (The percentage of time UAVs maintain communication links with the GCS) [5], and target handling efficiency (Detection and information relay times). These metrics are measured in teams of 4 to 20 UAVs and in different environmental conditions, providing both practical insights and mathematical substance that future researchers can build upon.

The progression of our research reflects the evolution in multi-UAV path planning solutions, moving from static optimization through basic reinforcement learning to advanced adaptive systems. This systematic development not only advances the technical capabilities of multi-UAV systems but also provides valuable implementation details into the integration of MOO with RL, creating a comprehensive resource for both researchers and practitioners in the field.

1.3 Thesis Organization

The remainder of this thesis is structured to systematically explore and develop the proposed solutions, as outlined below:

- **Chapter 2**: *Literature Review* – This chapter provides a comprehensive review of existing methodologies in multi-UAV systems, including traditional optimization techniques, reinforcement learning approaches, and evolutionary algorithms. It identifies gaps in the current research and sets the foundation for the proposed contributions.

- **Chapter 3**: *Multi-Objective Path Planning* – We propose a multi-UAV path planner that jointly optimizes area coverage time and connectivity among the UAVs.

A novel connectivity metric is introduced, which includes not only the percentage connectivity of the UAVs to the GCS but also the maximum duration of consecutive time that the UAVs are disconnected from the GCS. To solve this optimization formulation, we employ a multi-objective evolutionary algorithm with novel operations. Our solver is tested on single, two, and many-objective path planning problems, and the Pareto-optimal solutions are compared to benchmark weighted-sum based solutions. The results demonstrate that, unlike weighted-sum methods which provide a single solution based on prior user information, the proposed evolutionary multi-objective optimizers offer a diverse set of solutions that cover a range of mission time and connectivity performance, illustrating the trade-off between these conflicting objectives. This allows the end-user to choose the best path solution based on mission priorities during operation.

- **Chapter 4**: *Reinforcement Learning-Based Dynamic Path Planning* – We introduce a dynamic path planner that utilizes a multi-agent reinforcement learning (MARL) model with novel reward functions tailored for multi-UAV SAR missions. A mission environment is designed where a multi-UAV team covers an area to detect randomly distributed targets and inform the GCS by continuously forming relay chains between the targets and the GCS. The training procedure of the agents incorporates a CNN that processes images representing trajectory histories and connectivity states of each environment entity, such as UAVs, targets, and the GCS. Agents take actions and receive feedback from the environment until the mission is complete. The model is trained across multiple missions with randomized target locations. Results indicate that the trained model effectively generates mission plans that enable the multi-UAV system to search the area efficiently while dynamically forming relay chains. The proposed dynamic method achieves up to a 45%

improvement in total detection and mission times compared to a pre-planned optimized path planner.

- **Chapter 5**: *Advanced Reinforcement Learning Techniques for Heterogeneous Missions* – This chapter presents a novel MARL technique utilizing PPO to coordinate a UAV team in search and rescue operations, where multiple targets with different connectivity requirements are present. The proposed approach integrates a dynamic reward system that effectively manages the trade-off between exploration, target identification, and network connectivity. The model is trained for a fixed target type and area size, but is evaluated for scenarios with different numbers of UAVs, diverse target configurations, and mission priorities, including target search and information updates, illustrating its adaptability and versatility in comparison to common methods.

- **Chapter 6**: *Conclusions and Future Work* – The final chapter summarizes the research findings, discusses their implications, and outlines potential directions for future research to further advance the field of multi-UAV systems.

2. LITERATURE REVIEW

Significant advancements have been observed in the field of UAV systems over the past few decades. In order to provide a baseline for our work, this chapter will go through several works from existing literature relevant to our work. The review is structured into several sections, each focusing on distinct categories of approaches: traditional methods, SOO methods, MOO methods, learning-based systems. Each section includes detailed discussions and summary tables that encapsulate the key objectives, performance metrics, methodologies, and relevant studies.

2.1 Traditional Methods

Since the beginning of algorithmic programming, fundamental routing problems have caught the attention of many researchers. Initial investigations on routing problems began with applications of dynamic programming on the basic single-agent routing problem [15]. This foundational work led to other works such as the widely known Dijksra shortest paths algorithm [16]. Even though dynamic programming was sufficient for basic scenarios, the algorithm failed to provide efficiency under more complex scenarios. The researchers then began to solve these problems using heuristic algorithms, leading to the $A*$ path-finding algorithm [17].

Although a baseline for the basic single-agent path-finding problem was built over time, the need for multi-agent path finding algorithms was on the rise. This led to the proposition of the ALLIANCE architecture for multi-robot systems [18]. Multi-agent models require interaction between multiple agents. In general, this means there will be intersections and changing mission requirements. Therefore, after a preplanned path is generated, it must be governed by a more dynamic algorithm that balances the generated

routes in real time. As the need for such real-time models became significant, learning-based methods based on dynamic programming were proposed [19]. These models led the researchers to investigate the AI based structures further, along with heuristic optimization algorithms.

2.2 Optimization-Based Methods

Optimization-based methods enhance traditional approaches by utilizing more sophisticated mathematical models and algorithms to solve multi-UAV path planning problems. These methods can address both single-objective and multi-objective optimization problems, providing improved performance and adaptability. Evolutionary algorithms, such as Genetic Algorithm (GA) and Non-Dominated Sorting Genetic Algorithm (NSGA), play a significant role within this category by effectively handling multiple conflicting objectives. Table. 2.1 shows a set of different methods ranging from basic optimizers to multi-objective optimizers.

Table 2.1: *Comparison of Algorithms Based on Objectives.*

Algorithm	**Objective**
Dijkstra Modified [20]	Distance, Energy Cost
CBS [21]	Path Length, Collision Avoidance
GA [22]	Distance, Formation Control
PSO [23]	Energy, Time
MO-CBS [24]	Time, Path Risk
NSGA-II [25]	Distance, Safety
MOEA/D [26]	Risk, Distance, Formation
SPEA2 [27]	Energy, Time, Coverage
A* [28]	Distance, Communication Range
RRT-Connect [29]	Connectivity, Path Length
PSO-DTN [30]	Delay Tolerance, Distance

Early multi-UAV path planning research focused primarily on fundamental objectives like distance optimization and basic formation control. Zhang and Collins [23] introduced particle swarm optimization for energy-efficient search and rescue operations, while Chen and Li [20] enhanced this with a modified Dijkstra's algorithm incorporating battery constraints. The addition of formation requirements led Wang and Liu [22] to develop genetic algorithms that could maintain stable configurations during navigation.

As the field progressed, researchers began solving more complex multi-objective scenarios. Some researchers. [21] introduced a conflict-based search framework managing competing objectives such as fuel consumption and completion time. This evolution continued with [27] development of more evolutionary based solutions balancing energy consumption, time efficiency, and coverage area, while Atzmon et al. [24] enhanced computational efficiency through safe-interval path planning.

The most recent developments have focused on incorporating communication and networking constraints. Researchers [26] advanced the field further by addressing uncertain environments in heterogeneous systems. Uncertainty introduces entropy to the decision-making algorithms making the problem more realistic to real-life scenarios.

This progression from basic path planning to sophisticated multi-objective optimization with communication constraints demonstrates the field's evolution toward more practical and comprehensive solutions. It can be observed that communication quality became a concern as researchers accomplished the fundamental path planning problems.

2.3 Learning-based methods

Although optimization-based solutions provide a good baseline for the overall mission in the context of SAR, most multi-UAV missions include dynamically changing mission requirements. Such requirements include: New targets, relocation of existing targets, malfunctioning on UAVs along with many other unexpected events. For such

a system to be robust to these changes, the trajectories of each UAV must adapt the new requirements dynamically. Typically, this is modelled as a multi-agent system where there is an environment and a set of agents interacting with each other. Table. 2.2 shows a set of selected works.

Table 2.2: *Comparison of Machine Learning-Based Algorithms Based on Objectives.*

Algorithm	**Objective**
Hybrid Attention MARL [31]	Trajectory Optimization, Adaptation to Dynamic Environments
MADDPG [32]	Target Assignment, Path Planning
DRL with GNN [33]	Cooperative Control, Information Aggregation
DRL [34]	Energy Efficiency, Operational Effectiveness
MARL for Spatio-Temporal Coverage [35]	Area Coverage, Resource Allocation
Mean Field MARL with GAT [36]	Communication Efficiency, Task Allocation
Multi-Agent PPO [37]	Data Freshness, Quality of Service Optimization
DDPG for 3D Optimization [38]	Trajectory Optimization, Resource Allocation

In the context of multi-UAV missions, the dynamic nature of mission requirements necessitates robust solutions that can adapt to changing conditions. Recent advancements in RL have shown promise in addressing these challenges. Reinforcement learning, particularly in multi-agent settings, allows UAVs to learn optimal trajectories and strategies in environments characterized by uncertainty and variability.

The Hybrid Attention Multi-agent RL is a method that utilizes the generalization abilities of reinforcement learning to adapt dynamically to novel environmental situations [31]. This approach utilizes attention processes to analyze time series data, improving UAV responsiveness to evolving mission demands. The Multi-Agent Deep Deterministic

Policy Gradient (DDPG) method has been utilized to provide concurrent target assignment and path planning for many UAVs, showcasing its efficacy in dynamic environments [32]. These methods illustrate the application of reinforcement learning to enhance UAV trajectories in accordance with real-time alterations in mission parameters.

Another paper by Zheng et al. introduces a spatio-temporal coverage planning technique for cooperative search including several UAVs and sensors. This study presents the deployment of MARL in situations when environmental data is deficient or inaccessible, requiring a model-free strategy to enhance coverage and resource distribution [35]. The authors illustrate that through the utilization of MARL, a multi-UAV team can collectively acquire knowledge to enhance their trajectories, effectively balancing several objectives, including the maximization of area coverage and the minimization of mission duration.

Yang et al. enhance this subject by introducing a partially observable mean field multi-agent reinforcement learning framework utilizing graph attention networks for UAV swarms. This method enables the modeling of interactions among UAVs in a collaborative environment, enhancing the optimization of many objectives, such as communication efficiency and job allocation[36].

Ndiaye et al. concentrate on enhancing data freshness in UAV-assisted networks by employing a multi-agent proximal policy optimization framework optimizing for resource allocation. This research presents the issue as a mixed-integer nonlinear programming problem, aiming to decrease the overall age of updates while adhering to time and quality of service restrictions. The suggested method illustrates the efficacy of Multi-Objective Reinforcement Learning in solving for conflicting objectives in UAV operations [37].

Overall, we can observe the clear evolution from traditional path planning methods

to sophisticated learning-based approaches that can handle dynamic, multi-objective scenarios while maintaining critical communication constraints. This progression provides the theoretical foundation for our proposed framework presented in Chapter 3, where we present our MOO framework for optimizing multiple competing objectives in a computationally efficient evolutionary algorithm setup.

3. MULTI-OBJECTIVE COVERAGE AND CONNECTIVITY OPTIMIZATION FOR MULTI-UAV SYSTEMS WITH EVOLUTIONARY ALGORITHMS

3.1 Introduction

In this chapter, we consider a multi-UAV mission that requires exploration of an unknown area via onboard sensors and delivery of sensed data continuously to a GCS. While we do not specify any application, such requirements are to be expected for target search, data collection in Internet of Things (IoT) networks, surveillance or monitoring missions to name a few. To this end, *we formulate the multi-UAV path planning problem as a multi-objective optimization problem*.

We propose to *simultaneously optimize total area coverage time, percentage connectivity of the UAVs to the GCS throughout the mission, and maximum time interval that the UAVs are disconnected from the GCS*, while satisfying certain constraints including flight time and number of UAVs. The first objective aims to achieve the coverage task as fast as possible, whereas the remaining two objectives aim to enable continuous *multi-hop* connectivity of a high number of UAVs to the GCS, while avoiding isolated set(s) of UAVs in the network. The avoidance of isolated nodes is not explicitly considered in other works, but it is important for both safe operations and efficient use of resources especially when the drone network is sparse. Furthermore, while continuous sensed data delivery might not be necessary for all UAV missions and some delay can be tolerable, maintaining connectivity of the UAVs to the GCS is desirable, since it allows the GCS to keep track of the UAV and mission status and enables team behaviour via the resulting UAV-UAV connectivity.

Although we consider connectivity as an optimization parameter for continuous

sensed data delivery, as we discuss in [2], the data to be transmitted during a UAV mission may include mission command data (from the GCS); telemetry data from the UAVs to ground (and/or other UAVs); sensed data from the UAVs, etc. Therefore, even if, for instance, some sensed data can be delay-tolerant and can be downloaded once the mission is complete, the UAV status information such as GPS coordinates, battery level, speeds, altitude, etc. need to be known reliably at the GCS for safe operation. Therefore, multi-hop connectivity to GCS is very important. Furthermore, connectivity to GCS inherently enables team behaviour, since UAVs stay connected to each other and can be aware of their team members. Therefore, in an effort to enable continuous data delivery, UAV status tracking, and avoiding isolated UAVs, we choose percentage connectivity and maximum disconnected time as two of our optimization parameters.

The main solution approach used in multi-objective path planning in multi-UAV applications is weighted sum due to its simplicity. Weighted-sum method, in essence, converts the MOO problem to a SOO problem and provides a single optimum solution. However, it also requires prior knowledge about the end user's priority regarding the optimization objectives, which might not always be available. Furthermore, in cases where the objectives conflict with each other such as in this chapter, a set of diverse solutions might be more desirable than a single optimum. Therefore, for the solution of our *novel* MOO problem, we propose a multi-objective evolutionary algorithm based on NSGA. We introduce sampling, mutation, crossover, and repair operations suitable for our multi-UAV area coverage and connectivity mission and provide *Pareto-optimal* solutions to the MOO problem.

The formulation of the optimization problem and its implementation are done in such a way that any combination of objective functions (single or multiple) can be analyzed and further objective functions can be added. There are many solvers such as jMetal, vOptSolver, TEA, PyMOO, etc. for the MOO problem [39]. In this work, we use

the PyMOO [40] library for Python which is a multi-objective optimization-based library that contains open-source implementations of different algorithms.

We implement multi-UAV path planners with single, two, and many objectives. We provide weighted-sum solutions for the proposed objectives and a reinforcement-learning based mission time optimized model, as a benchmark. We compare the performance of our proposed algorithms in terms of their achieved objective function values; namely, percentage connectivity, total mission time, and maximum disconnected time. The SOO results illustrate the trade-off between coverage and connectivity. When only percentage connectivity is optimized, total mission time increases by 50%, whereas when total mission time is optimized, percentage connectivity is decreased and maximum disconnected times are increased by 50%. When coverage and connectivity are jointly optimized, total mission times are improved without degrading the connectivity significantly. When compared to weighted sum method with equal weights for each objective, the NSGA-based solutions provide at best a similar performance when two-objectives are jointly optimized. However, the weighted-sum solution falls within the upper and lower bounds of the set of solutions provided by NSGA-II and III when more than two objectives are simultaneously optimized. This implies that if there is no prior knowledge of user preferences regarding the objective functions, MOO-based solutions become more preferable, when more objectives are added to the optimization formulation.

Our main contributions can be summarized as:

- We propose a MOO formulation with *novel* connectivity and coverage objectives for multi-UAV path planning. We aim to optimize topological connectivity rather than network performance, since topological UAV-GCS connectivity and/or UAV-UAV connectivity to enable swarming is expected to be more relevant for applications such as surveillance, search and rescue, monitoring.

- We provide a multi-objective evolutionary algorithm framework to our optimization problem and propose sampling, mutation, crossover, and repair operations usable for *any* genetic algorithm.

- Using the proposed framework, we provide novel NSGA-II and NSGA-III based *Pareto-optimal solutions* with *varying number of decision variables* using the PyMOO library and compare their performance to benchmark weighted-sum and reinforcement learning based solutions with different levels of connectivity requirements. With a solution set rather than a single solution, we enable the end user to choose the path plan that fits the mission needs at the time of deployment or during the mission.

The remainder of the chapter is organized as follows. In Section 3.2, we provide the system assumptions, performance metric definitions, and mathematical model for our MOO problem. The proposed multi-objective evolutionary solution is presented in Section 3.3. Results are given in Section 3.4 and the chapter is summarized in Section 3.5.

3.2 System Model and Problem Formulation

The input and mathematical model parameters are defined in Table 3.1. The assumptions, optimization parameters, and mathematical model for our multi-objective optimization problem are given in the following.

3.2.1 Assumptions

The goal of the mission is to cover a given area with a multi-UAV team while simultaneously minimizing mission time and maximizing connectivity of the UAVs to the GCS. The number of UAVs is fixed.

The area of interest is assumed to be a square and it is divided into equally-sized,

disjoint cells, where the size of each cell depends on the ground sensing coverage area of the UAVs. The size of the length of each cell is denoted by $\mathbf{A}$. There are no obstacles or forbidden flight zones for the drones to navigate through. The UAVs fly at a fixed height from the ground. They are capable of waypoint flight and hovering. Each UAV is equipped with (i) a sensor, e.g., GPS, such that it knows its own position; (ii) a sensor such as camera to observe the environment; (iii) a wireless communication interface to exchange information with other UAVs and the GCS. The maximum flight times of the UAVs are limited to $\mathbf{T}_{max}$ given in Table 3.1.

Each UAV is expected to monitor visited cells with onboard sensors and work as relays for transmitting sensed data. The UAVs can directly communicate with other nodes in the network that are within their transmission range and nodes that are farther away can be reached over multi-hop links. Our previous works in [41, 42] which analyze path planning under sensing uncertainties and the impact of imperfect sensing on path planning performance, have shown that imperfect sensing affects different mTSP based path planners in a similar manner and the trends do not change. Therefore, in this chapter, we assume perfect sensing; i.e., if a drone visits a cell, the cell will be sensed in full without uncertainty. With regards to communication model, we assume disc model in accordance with our experimental work in [43], where we have proposed a three antenna-setup that is resilient to height and orientation differences between UAVs and leads to a radiation of omni-directional nature. We are currently in the process of incorporating networking performance under fading conditions into our path planning framework, based on our recent findings from ns3 simulations [44].

Each UAV starts their mission at the GCS at the same time and end their mission at the GCS. They traverse a sequence of cells, which constitute their path. The maximum number of steps that can be taken by a drone is denoted by K. Each step corresponds to one movement, where a drone goes from one cell to the next in their path. Drones can

Table 3.1: *Input and Mathematical model parameters.*

Input Parameters	**Definition**
$\mathbf{M}$	Number of drones
$\mathbf{r}_c$	Maximum transmission range
$\mathbf{V}$	Maximum drone speed
$\mathbf{G}$	Grid dimensions of area of interest
$\mathbf{P}$	Set of cells to be visited
$\mathbf{S}$	Maximum number of cells that can be traversed at a movement
$\mathbf{\Theta}_{max}$	Maximum disconnected time threshold
$\mathbf{\Delta}_{max}$	Maximum tour length threshold in terms of number of cells
$\mathbf{T}_{max}$	Maximum time that a mission can take
$\mathbf{A}$	Cell size in meters
$\mathbf{Y}$	Connectivity check period
Model Parameters	**Definition**
τ_n	Normalized total mission time
T_k^m	Total time required for drone m to move from step $k-1$ to step k in seconds
K	Maximum number of steps among all path sequences
Δ^m	Total sub-tour length of drone m in terms of number of cells
D	Set of distances between cells
$X_{i,j,k}^m$	Number of visits from cell i to j by drone m at step k
$C_{i,j,k}$	Connectivity between drone i and j at step k
$\Omega_{m,k}$	Connectivity to GCS of drone m at step k
Ψ	Percentage connected time throughout the mission
Θ_m	Maximum disconnected time of drone m
Ξ	Number of movements that cover more than S cells in a path sequence

move a distance of at most S cells at each movement due to their speed limit. Movements are stored in the visit matrix (X) given as:

$$X_{i,j,k}^m = \begin{cases} 1, & \text{if cell } j \text{ is visited after cell } i \text{ by drone } m \\ & \text{at step } k \\ 0, & \text{otherwise} \end{cases} \tag{3.1}$$

The drone movements are synchronized with each other; i.e., the drones with less cells to visit than K can hover over the same cell once they arrive at their next location. Therefore, the UAVs move onto their next location at the same time. Furthermore, the UAVs can hover to assist connectivity. We introduce the number of visits to a cell as a parameter to be used in future extensions of our work. For some missions, it might be necessary to visit some cells multiple times, e.g., in case of imperfect sensing or when updates about certain cells are desired. In this chapter, since we assume perfect sensing, we set the number of visits in non-consecutive path steps to a cell to one (hovering corresponds to visits to a cell in consecutive path steps).

3.2.2 Optimization Parameters

In this work, our goal is to fully cover (i.e., sense) a given area by a team of UAVs in the shortest time, while maintaining the connectivity of the UAVs to the GCS such that sensed data can be delivered to the ground. To this end, we define the following optimization parameters:

- *Total mission time*: Time taken to cover the area of interest.
- *Percentage connectivity*: Percentage of time the UAVs are connected to the GCS possibly over multi-hop links averaged over number of UAVs.
- *Maximum disconnected time*: The maximum duration of consecutive time interval a UAV is disconnected from the GCS.

Total distance taken by drone m, denoted by Δ^m, in terms of number of cells during the mission is formulated using the visit matrix X and distance matrix (D) as given in Eq. (3.2). $D_{i,j}$ corresponds to the distance between cells i and j. If there is a movement from cell i to j along the path of m at a given step k, then $X^m_{i,j,k} = 1$ and the

distance between them given by $D_{i,j}$ is added to the total distance.

$$\Delta^m = \sum_{i=1}^{\mathbf{P}} \sum_{j=1}^{\mathbf{P}} \sum_{k=1}^{K} X^m_{i,j,k} \, D_{i,j} \tag{3.2}$$

The time for drone m to complete step k of its generated path, (T^m_k), is given in Eq. (3.3), where **A** and **V** are the cell sizes in meters and maximum speeds of the drones, respectively. The step k will be completed when every drone arrives at their next position. Each drone can hover or move for the maximum amount of time required for any of the drones to arrive at their next position.

$$T^m_k = \begin{cases} \frac{\mathbf{A}}{\mathbf{V}} \, X^m_{i,j,k} \, D_{i,j}, & \text{if } X^m_{i,j,k} = 1 \text{ at step } k \\ 0, & \text{otherwise} \end{cases} \tag{3.3}$$

The objective function, i.e., the normalized total time taken to cover the area, is calculated using Eq. (3.4), where $\mathbf{T}_{max}$ is the maximum time a mission can take.

$$\tau_n = \frac{1}{\mathbf{T}_{max}} \sum_{k=1}^{K} \max_{\forall m \in \mathbf{M}} T^m_k \tag{3.4}$$

We assume that if drones are at most r_c cells apart, they are connected. Information is exchanged between drones and GCS over a multi-hop relay chain which can be formed if and only if a number of connected drones are positioned such that at least one of them is connected to the GCS.

Our aim is to maintain a relay chain at each step k for each drone. At a given step k, network of drones are modeled as a graph G containing GCS and the drones as vertices. There is an edge between any two vertices, if the distance between the corresponding nodes is less than r_c. Binary connectivity matrix (C) at k is then defined as the adjacency matrix of G given in Eq. (3.5):

$$C_{i,j,k} = \begin{cases} 1, & \text{if drone } i \text{ is connected to drone } j \text{ at } k \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

After C is computed, the indirect (multi-hop) or direct (single-hop) connectivity of each drone to the GCS is formulated as ($\Omega_{m,k}$) for each step k. The GCS is expressed as a node with index -1. The multi-hop connectivity values are obtained from undirected connection graphs generated by connected drone pairs. Eq. (3.6) shows the formulation of connectivity to GCS matrix (Ω):

$$\Omega_{m,k} = \begin{cases} 1, & \text{if } m \text{ is connected to } GCS \text{ at } k \\ 0, & \text{otherwise} \end{cases} \tag{3.6}$$

Then, percentage connectivity (Ψ) is calculated by averaging the values of Ω over number of drones and number of steps as given in Eq. (3.7). This value will be our second objective function and we aim to maximize it.

$$\Psi = \frac{1}{K}\frac{1}{\mathbf{M}}\sum_{k=1}^{K}\sum_{m=1}^{\mathbf{M}}\Omega_{m,k} \tag{3.7}$$

Finally, disconnected times for each drone, denoted by the 1x$\mathbf{M}$ vector (Θ), is calculated by taking the maximum number of consecutive disconnection occurrences from the GCS for each drone as shown in Algorithm 1. The optimization parameter is the normalized maximum disconnected time.

3.2.3 Mathematical model for the multi-objective optimization problem

Objective functions to be minimized are the normalized total mission time and maximum disconnected time, whereas percentage connectivity is to be maximized. Our implementation of the evolutionary algorithm based solver allows optimization of any combination of these objectives. However, we will present results on a subset of formulations summarized in Table 3.2 in addition to single objective optimization. First two rows of Table 3.2 formulates the objective functions O_1 and O_2 for weighted sum method with two and three objectives, respectively, where each objective is assigned equal weights.

Algorithm 1 Disconnected Time Computation

Input : Ω
Output : Θ

Initialize Θ
for $m \in$ **M do**
    Initialize $disconnected$
    for $k \in K$ **do**
        $disconnected \leftarrow disconnected + \ \Omega_{m,k}$
        if $\Omega_{m,k} = 1$ or $k = K$ **then**
            if $disconnected > \Theta_m$ **then**
                $\Theta_m = disconnected$
            end if
            $disconnected \leftarrow 0$
        end if
    end for
end for
Return Θ

Table 3.2: *Objective functions.*

Type	**Formulation**
Weighted sum of three objectives with equal weights	$O_1 = \frac{1}{3}(\tau_n + \max(\Theta) - \Psi)$
Weighted sum of two objectives with equal weights	$O_2 = \frac{1}{2}(\tau_n - \Psi)$
Two objectives	$O_{3,1} = \ \tau_n$ $O_{3,2} = \ 1 - \Psi$
Many objectives	$O_{4,1} = \ \tau_n$ $O_{4,2} = \ \max(\Theta)$ $O_{4,3} = \ 1 - \Psi$

The last two rows are MOO formulation of our optimization problem, where two or more objectives are simultaneously optimized. The subscripts i in $O_{3,i}$ and $O_{4,i}$ correspond to the different objectives. While the weighted sum method provides a single optimum solution, MOO formulation returns a set of Pareto-optimal solutions. The objectives that are not to be optimized are set as constraints in the corresponding mathematical formulation.

The objectives need to be accomplished subject to the following constraints:

$$
\begin{aligned}
\text{Minimize} \quad & O_i \\
\text{s.t.} \quad & \sum_{j=1}^{P} X_{i,j,k}^{m} \in (0,1), \forall i \in \mathbf{P}, \forall m \in \mathbf{M}, \forall k \in K, \\
& \sum_{m=1}^{\mathbf{M}} \sum_{i=1}^{\mathbf{P}} \sum_{j=1}^{\mathbf{P}} \sum_{k=1}^{K} X_{i,j,k}^{m} = ||\mathbf{P}||, \\
& \Xi = 0 \\
& \max \Delta \leq \mathbf{\Delta}_{max} \\
& \max \Theta \leq \mathbf{\Theta}_{max} \\
& \tau_n \leq \mathbf{T}_{max} \\
& \text{Number of Drones} = \mathbf{M}, \\
& \text{Base Station} = \mathbf{M}_{-1}, \\
& \text{Maximum Transmission Range} \leq \mathbf{r}_c, \\
& \text{Area of Interest Size} = \mathbf{G}, \\
& X_{i,-1,k}^{m} = 1, \forall m \in \mathbf{M}, \exists i \in \mathbf{P}, \\
& X_{-1,i,k}^{m} = 1, \forall m \in \mathbf{M}, \exists i \in \mathbf{P},
\end{aligned}
\tag{3.8}
$$

The first two constraints in the formulation implies the drones visit each cell once. We assume that speed is violated when the distance between consecutive movements of a drone is greater than S. As such, we define the number of speed violations (Ξ) as:

$$
\Xi = \sum_{m=1}^{\mathbf{M}} \sum_{i=1}^{\mathbf{P}} \sum_{j=1}^{\mathbf{P}} \sum_{k=1}^{K} (S - X_{i,j,k}^{m} D_{i,j} \leq 0) \tag{3.9}
$$

Third constraint then enforces the paths to have zero speed violations to prevent intersections between sub-paths. Fourth constraint limits the maximum tour length amongst all drones to prevent a single drone from covering most of the area alone. Fifth constraint limits the maximum disconnected time such that each drone gets connected to the GCS at

least once during the mission. The next constraints limit the number of drones available for the mission, define the base station and the maximum transmission range and area of interest. Final two constraints enforce the drones to start and end their missions at a fixed starting point (i.e., the GCS).

3.3 Proposed Evolutionary Algorithm Solutions to the Optimization Problem

3.3.1 Preliminaries

General form of MOO problem is given as [45, 46]:

$$\begin{aligned} &\text{min/max} && f_m(\mathbf{x}), && m = 1, 2, ..., M \\ &\text{subject to} && g_j(\mathbf{x}) \geq 0, && j = 1, 2, ..., J \\ & && h_k(\mathbf{x}) = 0, && k = 1, 2, ..., K \\ & && x_i^{lower} \leq x_i \leq x_i^{upper} && i = 1, 2, ..., n \end{aligned} \tag{3.10}$$

where there are M objective functions $f_m(\mathbf{x})$, J inequality and K equality constraints, given by $g_j(\mathbf{x})$ and $h_k(\mathbf{x})$ functions, respectively. Solution $\mathbf{x}$ is a vector of n decision variables with each decision variable x_i restricted to a value within a lower and an upper bound.

In SOO, the optimality of a solution with respect to other solutions can be determined by comparing the objective function values. However, in MOO it might not be possible to obtain a single global optimum solution that simultaneously optimizes all objectives. Therefore, the goodness of a solution in MOO can be determined by its dominance. In MOO solution methods, it is common to determine the Pareto-optimal set or non-dominated solutions generating a Pareto Front. Pareto front corresponds to the set of solutions where none of the objectives can be improved without sacrificing the other objectives. To this end, in MOO, each solution is compared to others on different objective

values and a distance value is computed. The distance value is used to select some of the solutions to be mated with each other. At each generation, a new solution set is chosen. The desired result is a Pareto optimal front of solutions with varying advantages on different objectives. Users can have the freedom to choose a preferable solution from this optimal set as opposed to the single optimum solution the SOO approaches can provide.

Depending on when the user preference can be provided, MOO approaches can be classified into three groups [39]: i) a priori, ii) a posteriori, and iii) interactive, where the weight or priority of the objective functions are known before optimization, after a set of non-dominated solutions are provided, and during the optimization, respectively. In this work, we focus primarily on a posteriori approaches, as will be described below. In a posteriori or interactive approaches, to assist the users while choosing a solution, a multi-criteria decision making (MCDM) method can be used by giving a set of weights that can set a trajectory to a single solution on the Pareto optimal front. An implementation of such a method is provided within the PyMOO library [40] and will be part of our future work.

In the solution of MOO problems, mathematical programming based scalarization as well as nature inspired meta-heuristic methods are frequently used. Among the scalarization methods are weighted sum, ϵ-constraint, goal programming and weighted metric [47]. One of the most commonly used methods is weighted-sum, which converts the MOO problem to a SOO problem by multiplying each objective with a user-provided weight and optimizing their sum [48]. The weights assigned to each objective depends on the respective importance of an objective. Due to their ease of use and ability to reach an optimum, scalarization methods have been attractive among researchers. However, their *need for prior knowledge regarding the weights of the objectives is a disadvantage*. Therefore, nature-inspired meta-heuristic methods such as Multi-Objective Evolutionary Algorithm (MOEA) are commonly used for MOO problems [45]. MOEAs aim to

find a Pareto-optimal solution set in a single run and are generally based on three design paradigms [39]: i) Pareto-based MOEAs are based on a two-level ranking and two most popular approaches are nondominated sorting genetic algorithm (NSGA-II) and strength Pareto evolutionary algorithm (SPEA); ii) Indicator-based MOEAs use a performance indicator for a set of solutions, such as hypervolume or R2 indicators; iii) Decomposition-based MOEAs, on the other hand, decomposes the MOO problem into several subproblems and uses a scalarization approach for each subproblem. MOEA/D (multi-objective evolutionary algorithm based on decomposition) and NSGA-III are well-known examples of this approach.

As summarized in Chapter 2, many of these deterministic and evolutionary approaches and others (Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Variable Neighborhood Search (VNS),Tabu Search (TS)) have been applied to SOO and MOO problems for various multi-UAV applications. Evolutionary algorithms can converge fast to a highly diverse and good quality near-optimal set of solutions [49]. They can be easily scalable to many objectives (3 and above). In contrast to deterministic methods that may return single optimum solutions given prior input from the user, evolutionary methods can provide a set of solutions that constitute a Pareto front and enable the end-user to observe and evaluate trade-offs between conflicting constraints. Therefore, in this work, we choose MOEA based solutions to the proposed problem formulation. In particular, we will build our solution on NSGA-II and NSGA-III due to their speed and scalability in terms of the number of objective functions. Further, as a benchmark, we propose a weighted sum solution, where the MOO problem is converted to an SOO as shown in Table 3.2.

In the following, we will first summarize our proposed solver and then describe in detail the modifications we have done to the GA operators to adapt to the proposed MOO problem.

3.3.2 Workflow

To solve the single, multi-, and many-objective path planning problems, we propose to use evolutionary algorithms based on GA, NSGA-II [50], and NSGA-III [51]. We implement our own sampling, crossover, mutation, and repair functions customized for multi-drone path plans.

Figure 3.1: *Workflow of the proposed solver.*

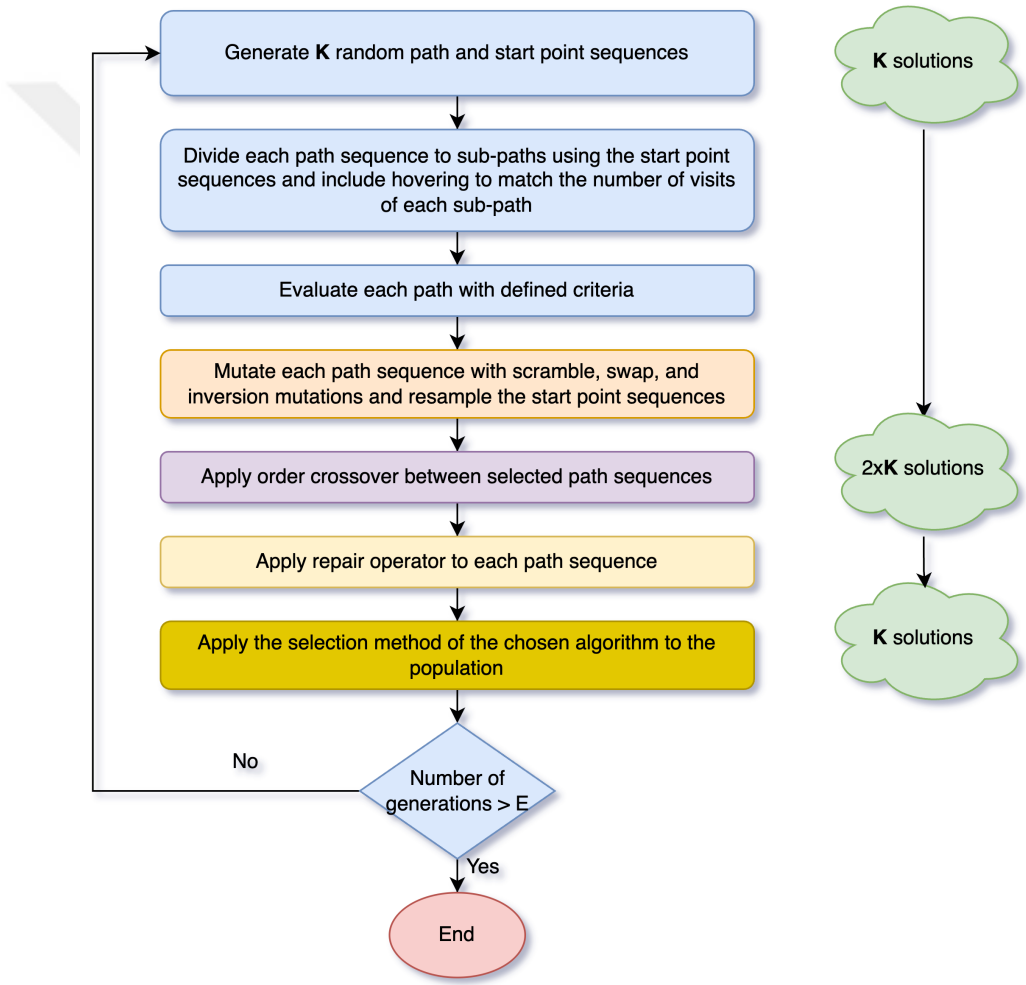


Figure 3.1 shows the workflow of our evolutionary solver. First, **K** random paths covering all cells are generated, forming a population. Each path sequence is divided

Table 3.3: *Evolutionary Algorithm Parameters.*

Algorithm Parameters	**Definition**
$\mathbf{K}$	Population size
$\mathbf{E}$	Number of generations
μ_{swap}	Probability of swap mutation
$\mu_{inversion}$	Probability of inversion mutation
$\mu_{scramble}$	Probability of scramble mutation
$\mathbf{Num_{ref}}$	Number of reference points
$\mathbf{O}$	Number of objectives

into $\mathbf{M}$ equal-sized sub-paths corresponding to individual drone paths. During evaluation, based on the distances between consecutive cells for each sub-path sequence, the movements of the drones are determined. Then, sub-path for each drone is interpolated with respect to the sub-path that has the highest number of movements using forward flat interpolation. This introduces hovering to the sub-paths; i.e., the drones that fly a shorter distance at a given step hover at their next position until all drones arrive at their next position in their path sequence. Due to interpolation, each drone has the same number of steps in their sub-path sequences. Then the sub-paths are evaluated using the objective values formulated in Eq. (3.4), Eq. (3.7), and Algorithm 1.

After evaluation, genetic operations detailed in Section 3.3.3 are applied to the solutions. We first combine swap, inversion, and scramble mutation operations. Using the probability parameters given in Table 3.3, a percentage of the solutions are mutated to create variance between the population and diverge from local optima. Then, parents are selected for the crossover operation where two feasible solutions are mixed with each other forming new solutions. These new solutions are repaired using our proposed repair operator to remove path irregularities such as large jumps.

The population is then sent to the selection stage where the solutions are sorted. Since there are multiple objectives, the algorithm finds solutions which are non-dominant

to each other such as a solution with high percentage connectivity, but high total mission time compared to another solution with low percentage connectivity but low mission time. These non-dominated solutions form a Pareto optimal front. We use crowding tournament selection, while implementing NSGA-II which selects individuals based on their crowding distance computed by their collective objective values [50]. NSGA-III uses reference points to select individuals that have distances close to pre-selected points. Our algorithm generates new solutions using the explained mutation and crossover functions to be compared with each other.

After the selection stage, the generation number is checked, if the number of generations has not exceeded **E**, each solution is evaluated again, and the procedure repeats itself. Otherwise, we get a list of non-dominated solutions on a scatter map and observe the Pareto optimality. Finally, each cell-sequence based non-dominated solution is interpolated to compute the locations of drones at each second.

3.3.3 Optimizer

For the design and implementation of the optimizer, we use the open source PyMOO library developed with Python [40]. The library contains the implementations of many of the popular single and multi-objective optimization algorithms. Table 3.4 shows the algorithmic complexities of the used algorithms considering the structure of the algorithms [50], [51] and the objective function related connectivity and mission time computations. Complexities are affected by the number of objective functions ($\mathbf{O}$) and population size ($\mathbf{K}$). Additionally, NSGA-III also has the $\mathbf{Num_{ref}}$ parameter denoting the number of reference points chosen for generating the Pareto optimal fronts at the end of the iterations. Each evaluation goes over each cell and drone for distance computations and goes over each drone two times for connectivity matrix computations.

We implemented our own solution structure for both optimizing path sequences

Table 3.4: *Complexity of the path planners.*

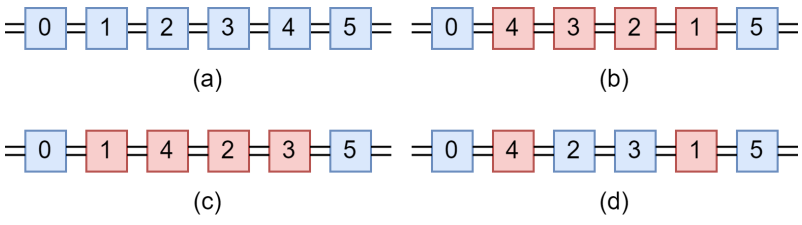
Algorithm	**Estimated Time Complexity**
NSGA-II	$O(\mathbf{K} \times \log(\mathbf{K}) \times \mathbf{O} \times (\mathbf{M} \times \mathbf{P} + \mathbf{M}^2))$
WS	$O(\mathbf{K} \times \mathbf{K} \times \mathbf{O} \times (\mathbf{M} \times \mathbf{P} + \mathbf{M}^2))$
NSGA-III	$O((\mathbf{K} \times \mathbf{K} + \mathbf{K} \times \mathbf{Num_{ref}}) \times \mathbf{O} \times (\mathbf{M} \times \mathbf{P} + \mathbf{M}^2))$

and path division points to split the path sequence to multiple drones. Then, we modified and extended each genetic operator to work with our solution structure. The proposed genetic operators are implemented as follows.

Given the number of samples **K**, the sampling operator initializes a population object. Each member solution of the population consists of a path sequence and a set of start points with length **M**. The path sequences are random permutations of the **P** cells. The start points are randomly generated in two stages. First, a randomly ordered binary list of length **P** is generated with **M** 1s and **P-M** 0s. Then, the 1 valued indices are taken to fill a start point list. Finally, each path sequence and start point pair is added to the population object.

The proposed mutation operator applies inversion, scramble, and swap mutations [52] to each path sequence and randomly change a value in each start point list as shown in Figure 3.2.

Figure 3.2: *(a) original sequence, (b) inversion mutation, (c) scramble mutation, (d) swap mutation.*



Crossover operation merges the surviving solutions to generate new, possibly better solutions. We apply this procedure only to the path sequences while keeping start

point values constant. For merging paths, we use the order crossover method [53] which takes a random interval in a sequence and changes that interval with the value order of that interval on the second sequence. This crossover method preserves the permutations and changes the order of cells in both sequences. Both crossover and mutation operations may lead to infeasible jumps between movements since the relative position of cells are not considered during these operations. To fix these solutions, a novel repair operator is implemented.

Algorithm 2 Path Repair

Input : Initial path sequence $oldPath$
Output : Repaired path sequence $newPath$

1: Initialize $newPath$
2: Append $oldPath_0$ to $newPath$
3: **while** $len(newPath) <$ **P do**
4: $cell \leftarrow oldPath_0$
5: Remove $oldPath_0$ from $oldPath$
6: $interpolatedCells \leftarrow$ interpolated list of cells between $cell$ and $cellPrev$
7: **if** cell not in $newPath$ **then**
8: **for** $cellMid \in interpolatedCellList$ **do**
9: **if** $cellMid \notin newPath$ **then**
10: Append $cellMid$ to $newPath$
11: **end if**
12: **end for**
13: Append $cell$ to $newPath$
14: **end if**
15: $cellPrev \leftarrow cell$
16: **end while**
17: Return $newPath$

Algorithm 2 shows the path repair procedure for a path sequence. This algorithm generates a new path sequence by taking the visited cell pairs (points of the grid) and interpolating each pair (line 6). For instance, if a movement from coordinate (0,0) to (2,2) has occurred, then point (1,1) is added between them if it has not been added before (lines 7-10). This operation decreases the number of speed violations and, consequently, decreases the number of generations required for the algorithm to find feasible solutions.

3.4 Results and Discussions

In this section, simulation results for the proposed path planning algorithms are provided for an area of 400mx400m with cell sizes of 50mx50m, which corresponds to a grid of size 8x8 cells. Transmission range is assumed to be $\sqrt{8}$ cells (i.e., $\sim$140m). The tests are run for a various number of drones. When providing the results for the multi-objective path planners, *all generated Pareto optimal results are provided to indicate the diversity of the obtained solution sets* in comparison to single solutions provided by the weighted-sum formulations. The set of solutions are indicated by shaded-regions in the figures.

For the evolutionary algorithms, the number of generations, **E**, and population size, **K** are set to 4000 and 100, respectively. These values are chosen after observing the quality and the generation time of the obtained solutions. The μ_{swap}, $\mu_{inversion}$, $\mu_{scramble}$ values are chosen as 0.3, 0.4, 0.3, respectively. A higher probability value is given to the inversion mutation, since this operation does not break and change sub-paths but instead inverts them, which leads to a lower number of speed violations. $\mathbf{Num_{ref}}$ value is chosen to match the population size for different number of objectives. The input parameter values used in our simulations are shown in Table 3.5.

The analyzed algorithms are as follows:

- SOO_{τ_n}, SOO_{Ψ}, SOO_{Θ}: Single objective solutions that correspond to minimum total mission time, maximum percentage connectivity, and minimum of maximum disconnected time, respectively.

- $WS_{\tau_n,\Psi}$, $WS_{\tau_n,\Psi,\Theta}$: Weighted Sum (WS) solutions that correspond to two and three equally-weighted objectives, respectively.

- $NSGA-II_{\tau_n,\Psi}$, $NSGA-II_{\tau_n,\Psi,\Theta}$: NSGA-II solutions that correspond to two and three simultaneous objectives, respectively.

Table 3.5: *Input parameter values.*

Input Parameters	**Values**
$\mathbf{M}$	[4, 8, 12, 16]
$\mathbf{r}_c$	$\sqrt{8}$
$\mathbf{V}$	2.5 m/s
$\mathbf{G}$	8 $\times$8
$\mathbf{P}$	Set of cells (1 to 64) in $\mathbf{G}$ to be visited
$\mathbf{S}$	$\sqrt{5}$
$\mathbf{\Theta}_{max}$	50 %
$\mathbf{\Delta}_{max}$	32 cells
$\mathbf{T}_{max}$	1000 seconds
$\mathbf{A}$	50 meters
$\mathbf{Y}$	4

- $NSGA-III_{\tau_n,\Psi}$, $NSGA-III_{\tau_n,\Psi,\Theta}$: NSGA-III solutions that correspond to two and three simultaneous objectives, respectively.

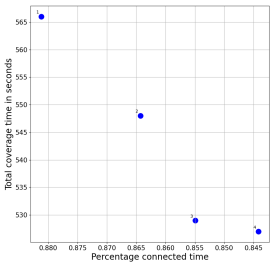
- RL_{τ_n}: In addition to the evolutionary algorithm solutions, we also provide a RL based model that optimizes total mission time, as a benchmark, which will be detailed in Chapter 4. For a fair comparison, the training time of the algorithm is limited to the runtime of the evolutionary algorithms. The RL based model is adopted from our recent path planner, which has been proposed for a dynamic target detection mission. Therefore, it can be used as a baseline for extending our proposed methods for more dynamic settings.

The simulations are run on an Intel Core i7-9750H Processor with 16GB of Random Access Memory (RAM). The computation times of the proposed methods do not differ significantly from each other and each simulation lasts approximately 20, 30, 35, 40 minutes for 4, 8, 12, 16 drones, respectively.

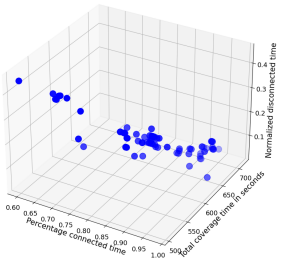
3.4.1 *Example path*

As a representative case, scatter plot of two objective values for NSGA-II$_{\tau_n,\Psi}$, when $M = 8$, indicating the percentage connectivity and total coverage time is given in Figure 3.3(a). The Pareto-front shows the trade-off between these two objectives. Figure 3.3 (b) shows a snapshot of the multi-UAV paths corresponding to solution 4 in Figure 3.3(a). The solid lines show the end-to-end paths of 8 drones. The current positions of the drones are marked as solid black circles, whereas the network graph generated based on the distances between the drones is shown by dotted black lines. Whole area will be covered at the end of the mission and all drones at this time instant are connected to each other.

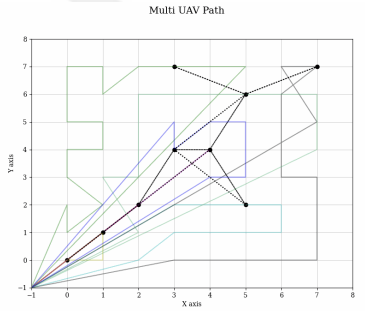
Figure 3.3: *Sample solutions for NSGA-II based path planners.*



(a) Scatter plot of two objective values of solutions optimized for $M = 8$ using NSGA-II$_{\tau_n,\Psi}$



(b) Scatter plot of three objective values of solutions optimized for $M = 8$ using NSGA-II$_{\tau_n,\Psi,\Theta}$



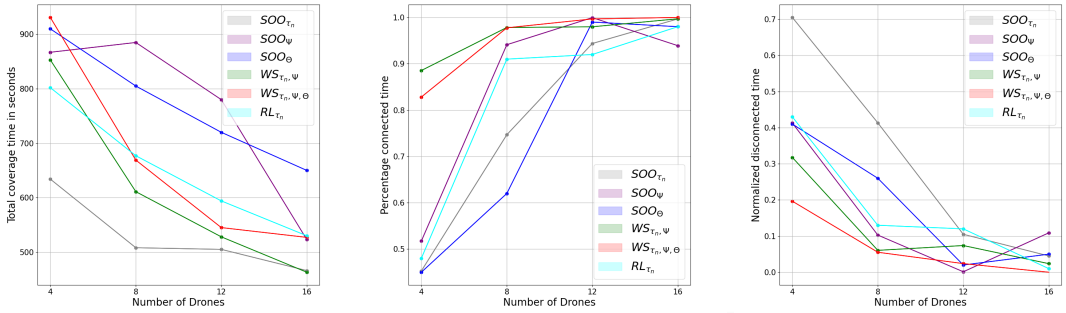
(c) Sample path snapshot solution number 4 of NSGA-II$_{\tau_n,\Psi}$

3.4.2 *Single-objective optimization based path planners*

We first provide performance in terms of total time, percentage connectivity and maximum disconnected time, when the multi-UAV paths are generated such that only one objective function is optimized using evolutionary and RL-based algorithms. As a comparison, we also provide the performance of weighted-sum GA for two equal weighted

objectives (i.e., total time and percentage connectivity) and three equal-weighted objectives (i.e., total time, percentage connectivity, and maximum disconnected time) as defined in Table 3.2. Figure 3.4(a) shows the total mission time versus number of drones. As expected, SOO_{τ_n} where total time is minimized leads to the best mission time. The SOO_{Ψ} requires 50% more time than the minimum time indicating that the drones might end up traversing longer paths or hovering in certain cells to assist connectivity of other drones to the GCS. The two and three-objective weighted sum methods lead to mission times slightly higher than the minimum, whereas RL_{τ_n} performs between the SOO and weighted sum approaches; better than connectivity optimized methods but worse than mission time optimized methods. Performance is expected to improve as the training time is increased. For low number of drones, there might be paths with slightly different mission times that lead to the same optimum weighted sum. As number of drones increase, the weighted sum methods approach SOO_{τ_n}. Figures 3.4(b) and 3.4(c) show the percentage connectivity and normalized maximum disconnected times of the SOO and weighted sum methods versus number of drones. Observe that the weighted sum methods, RL_{τ_n} and SOO_{Ψ} lead to up to 50% better connectivity than SOO_{τ_n} and SOO_{Θ}. This illustrates the trade-off between percentage connectivity and total mission time. When connectivity is optimized drones tend to stay together, whereas when total time is optimized the drones tend to spread in the area such that the coverage mission is accomplished faster. On the other hand, observe that optimizing maximum disconnected time does not necessarily lead to better percentage connectivity, whereas jointly optimizing percentage connectivity and disconnected time leads to better performance in terms of both parameters, especially when the number of drones is low. This is due to the fact that percentage connectivity optimization may lead to majority of the drones being connected to GCS, whereas some drones might be flying in an isolated manner majority of the time. When the disconnected time is also jointly optimized, this problem is solved.

Figure 3.4: *Comparison of total coverage times, percentage connected times, and maximum disconnected times of single objective models and weighted sum formulations with 2 and 3 objectives versus number of drones.*



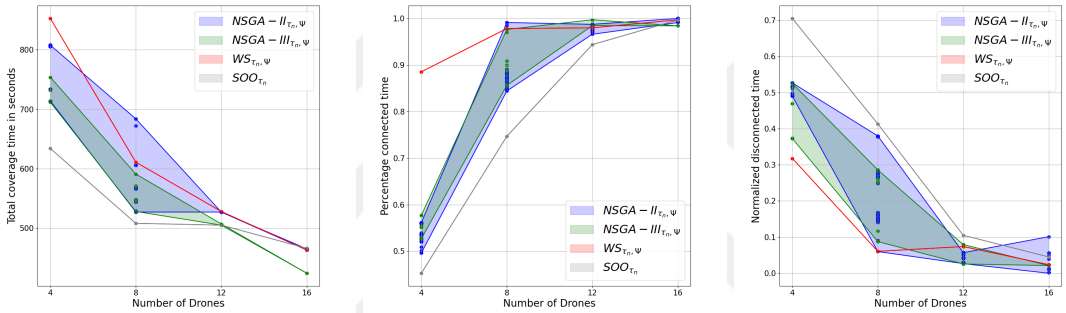
(a) (b) (c)

3.4.3 Two objective optimization based path planners

Next, we compare the performance of weighted sum, NSGA-II and NSGA-III algorithms, where the multi-UAV paths are generated such that the total mission time is minimized and the percentage connectivity is maximized. The weighted sum gives equal weights to both objectives, whereas NSGA-II and NSGA-III algorithms provide a set of solutions indicated by the shaded regions in Figure 3.5. For comparison, we also provide the SOO with time objective as a benchmark solution. SOO_{τ_n} leads to better mission times, and worse connectivity performance as expected. NSGA-II leads to a broader performance range for all parameters, whereas NSGA-III leads to a tighter solution set in terms of mission time. As number of drones increase, solutions provided by all schemes lead to narrower performance bounds. The weighted sum solution lies between NSGA-II and NSGA-III solutions in terms of mission time and sits at the lower and upper bounds of maximum disconnected time and percentage connectivity parameters of NSGA-II solutions, respectively. NSGA-III solution set corresponds to an acceptable connectivity

performance. Addition of the connectivity objective to the path plan optimization formulation leads to better connectivity throughout the mission, less isolated nodes, while keeping the increase in total mission time low, when number of drones is increased. This implies that two-objective optimization succeeds in finding better connected paths among the shortest time paths.

Figure 3.5: *Comparison of total coverage times, percentage connected times, and maximum disconnected times of 2 objective models optimized for total time and percentage connected time and single objective model optimized for total time versus number of drones.*

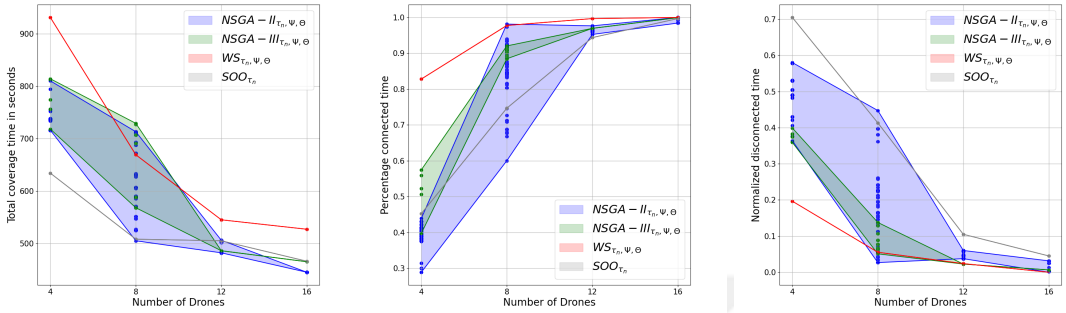


(a) (b) (c)

3.4.4 Many objective optimization based path planners

Next, we compare the performance of weighted sum, NSGA-II and NSGA-III algorithms, where the multi-UAV paths are generated such that the total mission and maximum disconnected times are minimized and the percentage connectivity is maximized. The weighted sum gives equal weights to all objectives, whereas NSGA-II and NSGA-III algorithms provide a set of solutions indicated by the shaded regions in Figure 3.6. The performance of SOO with time objective is also provided. While SOO_{τ_n} leads to better mission times, and worse connectivity performance, the total mission time degradation is less in comparison to two-objective optimization models with NSGA. NSGA-II and NSGA-III leads to a broad performance range for connectivity, when number of drones

Figure 3.6: *Comparison of total coverage times, percentage connected times, and maximum disconnected times of 3 objective models optimized for total time and percentage connected time and single objective model optimized for total time versus number of drones.*



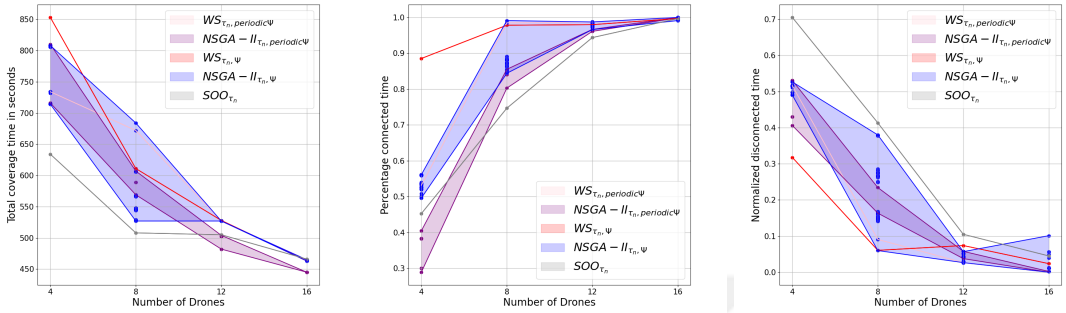
(a) (b) (c)

is low. As number of drones increase, solutions provided by all schemes lead to narrower performance bounds. Addition of the disconnected time objective to the path plan optimization formulation improves the performance of two-objective optimization models further. Two-objective optimization, while on average allows most of the nodes to stay connected to the GCS most of the time, it doesn't guarantee that some drones might fly in a disconnected manner. For instance, for $M = 4$, two-objective methods tend to create a communication chain of 3 drones, and one drone exploring the area. This leads to a high percentage connectivity but one drone is almost always disconnected from the GCS. With the addition of disconnected time objective, lower disconnected times are achieved even with lower number of drones.

3.4.5 Path optimization with periodic connectivity

Finally, to both illustrate the versatility of the proposed framework and investigate the impact of connectivity requirements further, we relax the assumption of continuous connectivity of UAVs to the GCS throughout the mission. To this end, we introduce periodic connectivity, where the UAVs are required to connect to the GCS every **Y** timesteps.

Figure 3.7: *Comparison of total coverage times, percentage connected times, and maximum disconnected times of single objective model optimized for total time, weighted sum and NSGA-II with 2 objectives for periodic and continuous connectivity formulations versus number of drones.*



(a) (b) (c)

This way, we enable free movement of drones during time steps between required connectivity periods. The formulation of percentage connectivity is accordingly given as:

$$periodic\Psi = \frac{\mathbf{Y}}{K}\frac{1}{\mathbf{M}}\sum_{k=1}^{\frac{K}{\mathbf{Y}}}\sum_{m=1}^{\mathbf{M}}\Omega_{m,k\mathbf{Y}} \tag{3.11}$$

Figure 3.7 compares the coverage times, percentage connected times, and normalized maximum disconnected times for WS and NSGA-II methods for continuous and periodic connectivity formulations, where for the latter the period is set to 4 time steps. Observe that the relaxed connectivity assumption, while eliminates the very high connectivity solutions, leads to a tighter solution set in terms of all parameters, which might help the user to choose a solution more easily. The total mission times drop below the continuous connectivity case, as expected, whereas connectivity performance of WS and NSGA-II are very similar to each other and sit in the middle of the NSGA-II solutions corresponding to continuous connectivity. If there is any prior knowledge regarding connectivity preferences of the user or if the preference changes, our proposed framework can be used to replan the UAV paths, accordingly.

3.5 Summary

In this chapter, we proposed a multi-drone path planner that jointly optimizes coverage time and connectivity among a team of drones, whose mission is to explore an unknown area via onboard sensors and deliver the sensed data to ground GCS. We proposed a novel connectivity metric and formulated the multi-drone path planning problem as a multi-objective optimization problem. To solve this problem, we proposed an NSGA-based multi-objective evolutionary algorithm with our own sampling, crossover, mutation and repair operations. We used our solver to test single, two and many objective path planning problems and compared our solutions to benchmark weighted-sum and RL-based solutions. We showed that when only coverage objective was optimized the connectivity was reduced by %50 and vice versa illustrating the trade-off between these metrics. We also showed that when the metrics were jointly optimized a range of solutions can be obtained with NSGA-II and NSGA-III based solvers, whereas weighted-sum solutions with equal weights given to objectives lie within the NSGA based solutions. NSGA-II based multi-objective optimizer provided a more diverse set of solutions to the user than NSGA-III. Finally, the addition of the third objective, namely, disconnected times led to better solutions in terms of all objectives. Our focus was on a pre-defined mission plan that could enable dynamic updates during the mission, if necessary, by incorporating connectivity into the mission plan. Our next focus is on extending this generic framework for more uncertain environments that would likely need a more dynamic response.

4. MAINTAINING CONNECTIVITY FOR MULTI-UAV MULTI-TARGET SEARCH USING REINFORCEMENT LEARNING

4.1 Introduction

Building upon the insights gained from our multi-objective optimization framework, we now turn our attention to developing a more dynamic and adaptive solution for multi-UAV path planning. While our previous approach successfully balanced coverage and connectivity objectives through evolutionary algorithms, it relied on pre-planned paths that may not adequately respond to the dynamic nature of search and rescue missions. During an SAR mission different types of targets might be of interest such as disaster victims, damaged buildings, damaged roads. Therefore, in this chapter, *we propose a dynamic path planner based on MARL* for our mission. The proposed *path planner dynamically generates paths* such that several uniformly distributed targets are detected and relay chains are formed from the target to the GCS until GCS is informed about each target. The actions of the UAVs are planned considering the current information states of all drones.

Dynamic decision-making enables quick responses to targets that appear at random cells. RL is a common method used for dynamic decision-making [54]. A simple RL system consists of an agent and an environment. Depending on the actions of the agent, the environment generates rewards and new states for the agent. For the multi-UAV SAR mission, the environment is a mission area with a GCS and hidden targets. The agents are UAVs with movement and sensing capabilities.

The proposed system gives each agent a limited number of actions that include

visiting a neighbor cell, visiting the closest unvisited cell while revisiting the cells in-between and hovering on the same cell to improve connectivity to the GCS. With the usage of a CNN and a DQN [55], the model is trained to give optimal decisions at any environment state using rewards that are computed from the states and actions of each agent. Novel reward functions are provided for the training to improve the model. These functions simultaneously reward the agents for visiting unvisited cells and sustaining connectivity to GCS.

For the implementation of the CNN, DQN and the agents, PyTorch [56] is used. For the design of the environment and other visualization components, Python libraries such as PyGame [57], NumPy [58], and MatPlotLib [59] are used. We tested the performance of our proposed path planner in terms of detection times, GCS notification, connectivity of the UAVs to the GCS and total distances traveled for a different number of UAVs and targets. We compare the performance with a GA based multi-objective optimizer, where the objective is formulated as a weighted sum of mission time and connectivity parameters. Our results show that the MARL model can improve the detection times of targets while maintaining the total mission time; therefore, the model dynamically changes the decisions for different target locations.

The key contributions of this chapter are:

- **Dynamic Decision Framework:** We develop a novel MARL-based path planner that enables real-time adaptation to changing mission conditions, moving beyond the limitations of pre-planned paths
- **Multi-Modal Reward Structure:** We introduce a comprehensive reward mechanism that simultaneously optimizes for area exploration and network connectivity, allowing for effective balance between competing objectives
- **CNN-Based State Processing:** We implement a CNN architecture for processing

Table 4.1: *System parameters.*

Parameters	Definition
$\mathbf{N}_d$	Number of drones
$\mathbf{N}_{tar}$	Number of targets
$\mathbf{r}_c$	Maximum transmission range
$\mathbf{V}$	Maximum drone speed
$\mathbf{G}$	Grid dimensions of area of interest
$\mathbf{P}$	Set of cells to be visited
$\mathbf{K}_{max}$	Maximum number of allowed action steps for a mission
$\mathbf{A}$	Cell size in meters

complex environmental states, enabling more sophisticated spatial reasoning and coordination between UAVs

The remainder of the chapter is organized as follows. In Section 4.2, we provide the system assumptions and definitions of performance metrics. The proposed MARL model is detailed in Section 4.3. The results are given in Section 4.4. The chapter is summarized in Section 4.5.

4.2 System Model

The system parameters are defined in Table 4.1. The area of interest **G** is divided into equally-sized, disjoint cells such that each cell corresponds with the ground sensing coverage area (**AxA**) of the UAVs. We consider multi-rotor UAVs which know their own positions and are capable of waypoint flying and hovering. We assume that UAVs fly at a given height from the ground. Each UAV is both a sensor that can observe the environment, and a network node with wireless interface, enabling data collection, multi-hop communication and information exchange among the UAVs and GCS. We assume the UAVs are equipped with an omni-directional antenna setup such as in [43] and hence, assume a disc model for communication ranges.

We further assume that if a drone visits a cell that contains a target, the target will

be detected without uncertainty. Sensor imperfections will be part of our future work. However, the interested readers are referred to our earlier work on path planning with sensor uncertainties in [41].

Drones start their mission at the GCS at the same time and return to the GCS once all targets are detected and connected to the GCS. They traverse a sequence of cells **P**, which constitute their path. Maximum flight time is indicated by the maximum number of cells visited by a drone denoted by K_{max}. The drone movements are synchronized with each other. At a given time step, drones can set their destinations to a distant unvisited cell. To sustain the synchronization for such cases, a move penalty value is computed using the distance between the destination and current position. The computation of the penalty is detailed in Section 4.3.

Performance metrics of interest are (i) *detection time*: time all targets are detected; (ii) *informed time*: time GCS knows of all targets; (iii) *percentage connectivity*: percentage of time UAVs are connected to the GCS averaged over number of UAVs and the mission time; (iv) *total travelled distance*: distance travelled by all drones till GCS is informed of all targets.

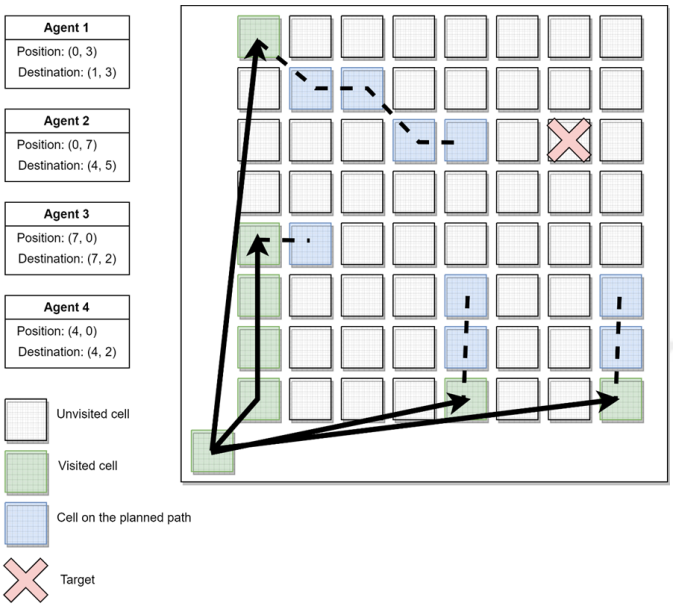
4.3 Multi-agent Reinforcement Learning Model

4.3.1 Agents and The Environment

As defined in Section 4.2, drones are tasked to detect and inform the base station about targets. To design a dynamic path planner for the defined mission, a MARL based model is proposed. In general, MARL models consists of a set of agents and an environment that the agents interact with [55]. An agent is a decision-maker that takes actions and gets feedback from the environment as rewards or penalties depending on the state transitions caused by the actions. As shown in Figure 4.1, each agent has positions and destinations. The destinations are determined by the actions. Targets are randomly

distributed over a grid of cells.

Figure 4.1: *Sample environment state. Solid lines show the trajectory histories of 4 drones and dashed lines indicate their upcoming movements.*

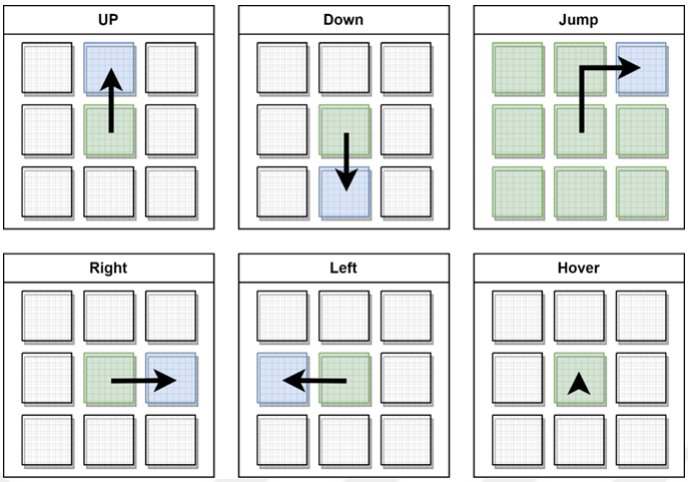


Possible actions for each agent are (i) movement to an adjacent cell (up, down, right, left in Figure 4.2), (ii) hovering over the same cell (hover in Figure 4.2), or (iii) choosing the closest unvisited cell as the destination and moving on a linear path of cells until the desired cell is reached (jump in Figure 4.2). Each action results in different rewards depending on the number of time steps required for the action to be completed or the number of hovering actions on the same cell, which will be discussed in Section 4.3.2.

The first cells to be visited after the start point are pre-defined for each agent. As shown in Figure 4.1, each agent starts from different positions. This decreases the chances of path intersections but also decreases connectivity between agents. An optimal selection of first visited cells will be part of our future work.

As shown in Figure 4.1, a jump action occurs, if the closest unvisited city is farther than 1 cell from the current position of the agent. In this case, the agent gets a movement penalty equal to the number of cells between destination and original position. Movement

Figure 4.2: *Action Types.*



penalty disables the agent's ability to make actions. During the penalized time steps, the environment autonomously moves the agent to the destination point while visiting the closest cell on the path until the destination is reached. This approach both eliminates large jumps that increase mission times and constrains the movement distance of each agent for each time step. At the resulting path plans, each agent moves at most one cell at each time step, unless they are released from or returning to the start point.

The mission workflow is described in Algorithm 3. At each action step, agents that are not on the move to a destination are allowed to make any of the defined actions in Figure 4.2. Each action is chosen when the turn of the agent comes to avoid actions towards the same destinations (lines 4-10). Actions are decided using epsilon-greedy algorithm (line 6) [55]. The chosen actions are applied to the environment (line 7). Network graph is formed with the current positions of the drones as vertices and with edges when the Euclidean distances between the drones is less than r_c. Adjacency matrices are then computed over this network graph (line 11). If an agent detects a target and it is not connected to GCS, other agents form relay chains and connect the target to GCS(lines 13-18). A sample relay chain is represented in Figure 4.3. The mission is considered to

be successful when GCS is informed about all the targets.

Algorithm 3 Episode Workflow

Initialize $drones$
while mission is not $done$ **do**
    for $drone \in N_d$ **do**
        if $drone$ has no pending visits and has no move penalty **then**
            $action \leftarrow$ get a selected action using epsilon-greedy algorithm [55]
            Apply $action$ and add new pending visits to $drone$
        end if
        Remove the first pending visit of $drone$ and add it to its path
    end for
    Compute the adjacency matrices using the drone and target positions using r_c
    for $target \in N_{tar}$ **do**
        if $target$ detected **then**
            $informer \leftarrow drone$ at the position of $target$
            Lock the position of $informer$
            if $informer$ is connected to GCS **then**
                Use breadth first search to find the shortest path from $target$ to GCS
            end if
        end if
    end for
    if all targets are detected **then**
        End mission and return to start point
    end if
end while

4.3.2 Rewards and Training

Figure 4.4 shows the workflow of the training process. At each step, agents take actions and new states are used to compute reward values. To evaluate both coverage and connectivity aspects of the mission, two reward functions are proposed. The sum of these reward values are used to update the deep learning model parameters of the agents.

The distance reward function has the following rules:

- Consecutive hovering actions on the same cell cumulatively penalize the next hovering. For instance, the fifth hovering action on the same cell results in a penalty value of 5.

Figure 4.3: *Sample relay chain formed by 6 drones, when 2 targets are detected.*

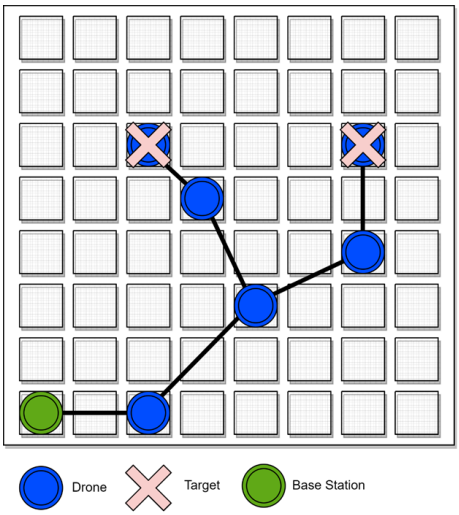
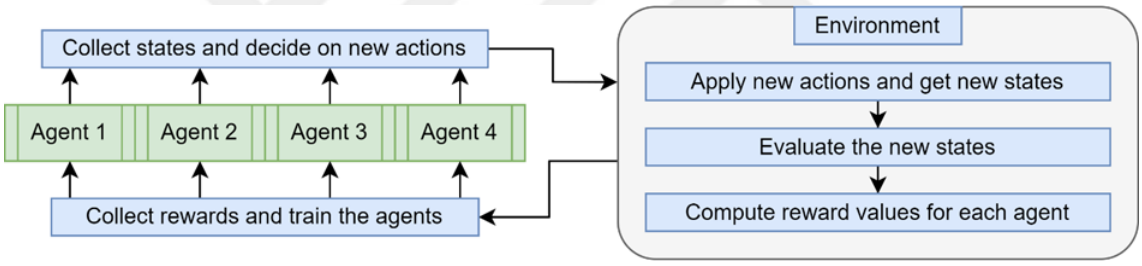


Figure 4.4: *Workflow of the multi-agent reinforcement learning model.*



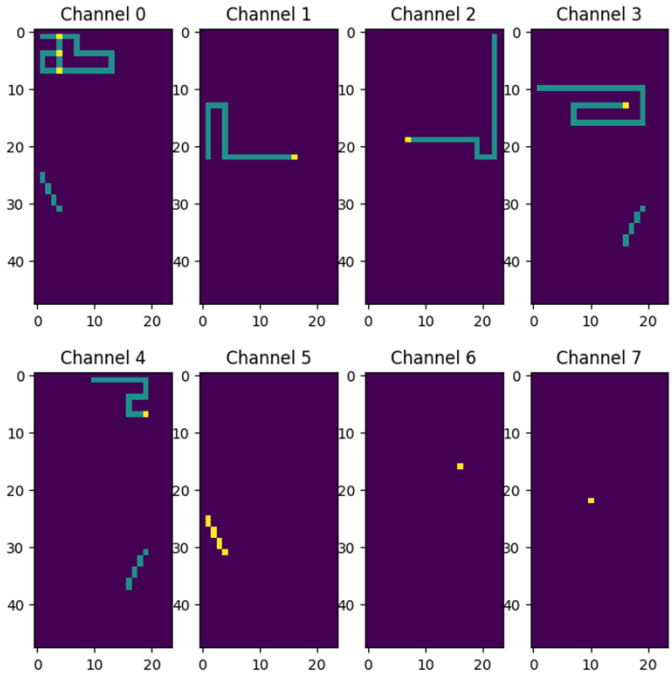
- Each jump action is evaluated by the number of unvisited cells on the path to the destination. Each penalized time step has a penalty value of 4 and each unvisited cell on the path has a reward value of 4.
- Each up, down, right, left action to an unvisited cell result in a reward value of 10.
- Each up, down, right, left action to a visited cell result in a penalty value of 5, unless all cells are visited at least once.

The connection reward function has the following rules:

- The reward value is the overall rate of connectivity of each target to the GCS multiplied by 10.

- At each new action step, the reward value is multiplied by the ratio of remaining action steps.

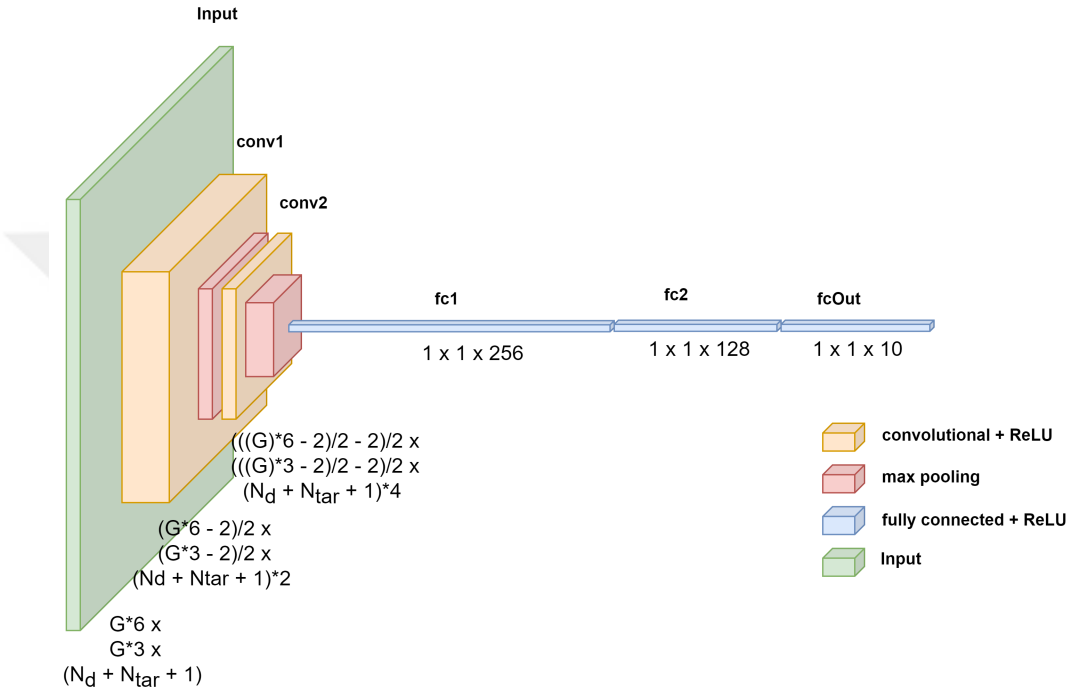
Figure 4.5: *Mission state images for a configuration with 5 UAVs and 2 targets. Channels 0-4 shows the path histories and connected components of UAVs 1-5. Channel 5 shows the connections between GCS and the UAVs. Channels 6 and 7 show the positions of the targets.*



Using the action histories of each agent and adjacency matrices, a multi-channel image is generated at each action step. Each channel expresses the coverage and connectivity states of each environment entity as a single channel image, where multiple visits are expressed as pixel magnitudes using logarithmic sums of the number of visits to each cell. The generated images shown in Figure 4.5 represent the states of 5 drones, the GCS, and the targets at the end of the mission. These images are used to train a CNN model for each agent at each new action and at the end of the mission collectively. For training, a CNN model and a DQN model are implemented using PyTorch [55, 56]. The implemented CNN uses 2 convolutional layers with 3x3 kernel sizes and 3 fully connected

layers (see Figure 4.6). Each convolutional layer doubles the number of channels. The output is then sent to a Q-learner module, where state changes are evaluated, and the reward value is used as a feedback for updating the parameters.

Figure 4.6: *CNN model for training.*



4.4 Results and Discussions

To test the proposed model, a simulation environment is designed, where size of the grid (G), number of drones (N_d), number of targets (N_{tar}), maximum number of action steps (K_{max}), transmission range (r_c) are inputs to the system. The values for input parameters are shown in Table 4.2. Google Colab environment and T4 GPUs are used for training the CNN and DQN models. For each simulation, models are trained for 100 episodes that can include successful and unsuccessful missions where even if GCSis not informed about all the targets, the mission ends when the number of action steps reaches K_{max}. The mission ends successfully if GCS is informed of all targets. Simulations are

Table 4.2: *Input parameter values.*

Parameters	**Values**
$\mathbf{N}_d$	$\{8, 12, 16, 20\}$
$\mathbf{N}_{tar}$	$\{1, 2, 4\}$
$\mathbf{r}_c$	$\sqrt{8}$ cells
$\mathbf{V}$	2.5 m/s
$\mathbf{G}$	8×8
$\mathbf{P}$	Set of cells (1 to 64) in $\mathbf{G}$ to be visited
$\mathbf{K}_{max}$	50
$\mathbf{A}$	50 m

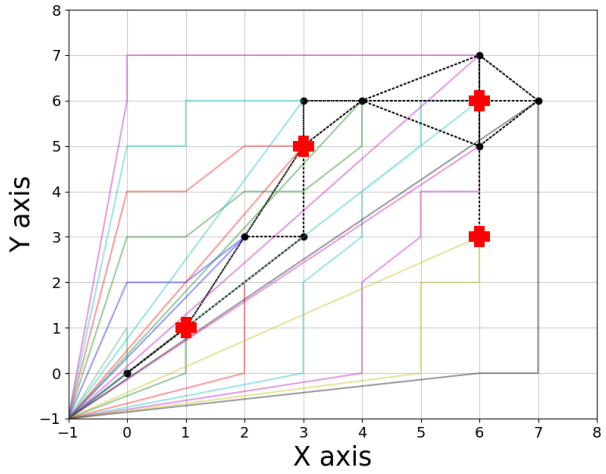
conducted using configurations with different N_d and N_{tar} values and results are averaged over 300 runs.

As a benchmark, we also implement a commonly used path planner based on a multi-objective optimizer, namely, weighted sum GA. This method jointly minimizes the area coverage time and maximizes percentage connectivity. Equal weights are given to the two objectives. Using this approach, we generate pre-planned paths for the drone team. The drones take-off and fly their pre-planned paths, until they know all targets are detected. At each time step, they sense their environment, if they detect a target, they inform all the nodes they are connected to over a multi-hop network. The mission is over, when the GCSknows of all targets.

Figure 4.7 shows a sample mission snapshot taken in the middle of the mission. The drones and targets are indicated with black dots and red crosses, respectively. The flown drone paths are shown by solid lines and the network graph among the drones are shown by black dotted lines. There is an edge between a pair of drones if the distance between them is less than the transmission range. The detected targets are connected to the base station through relay chains expanding to multiple targets.

Figures 4.8 and 4.9 show the detection and inform times versus number of drones for different number of targets. As expected, both times decrease with increased N_d. When compared to the pre-planned GA paths, detection times are lower in particular for

Figure 4.7: *Sample path snapshot for 8 drones (·) and 4 targets (+). Solid lines indicate the flown paths during the mission. Black dotted lines indicate the network graph.*



small N_d, indicating that the drone team learns to cover the search area in an efficient manner. The informed times are similar to detection times, indicating the detecting UAVs are connected to GCS at the time of target detection for both algorithms.

This can be further observed in Figure 4.10, which shows the percentage connected time for all cases. Observe that as N_d increases beyond 12 the UAV network becomes fully-connected throughout the mission. For smaller N_d, connectivity drops below 85% for MARL. GA leads to a higher percentage connectivity. These results illustrate the trade-off between search times and connectivity. While MARL has a lower detection time due to rewards given to visiting unvisited cells, its connectivity suffers due to spreading.

Finally, Figure 4.11 shows the total travelled times for MARL and GA versus N_d. The total distance of MARL is slightly higher. This indicates that the detection time difference is due to lengthier hovering in GA paths in an effort to provide better connectivity to the drones. This result also shows that optimizing search times and path lengths may lead to different performances depending on the other objectives of interest.

Figure 4.8: *Detection Times vs* N_d.

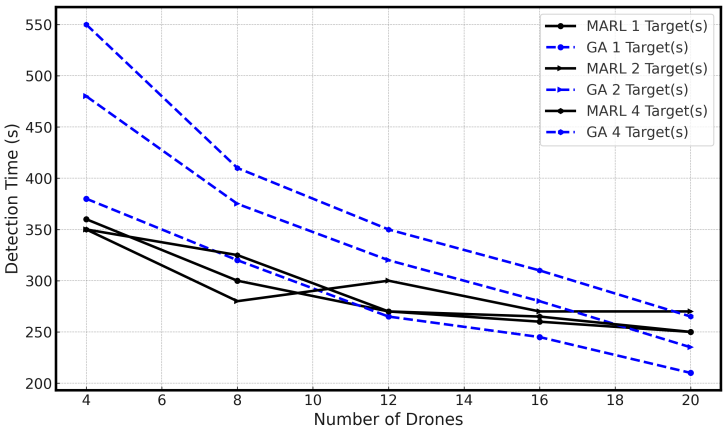
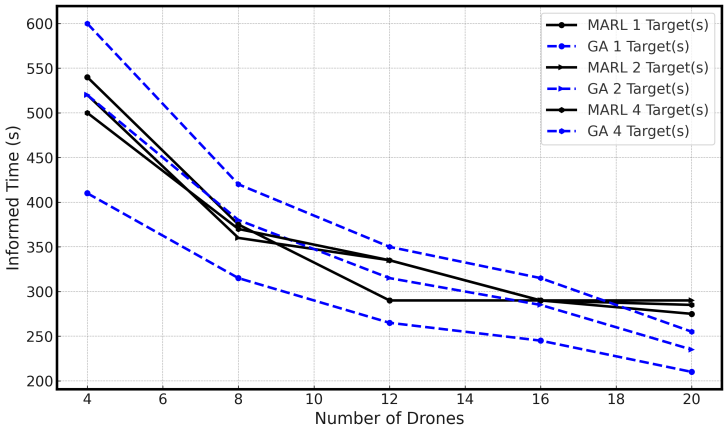


Figure 4.9: *Informed Times vs* N_d.



4.5 Summary

In this section, a dynamic path planner has been proposed for multi-UAV search and rescue missions, where multiple targets can appear at random locations and multiple UAVs need to coordinate their movements to establish relay chains between the GCS and the targets. The usage of MARL has provided a baseline for dynamic decision-making that is required for missions, where the requirements may change after drones have been released. The designed environment along with the reward functions has provided useful feedback to the agents and trained them using multi-channel images representing the state

Figure 4.10: *Percentage connected time vs N_d.*

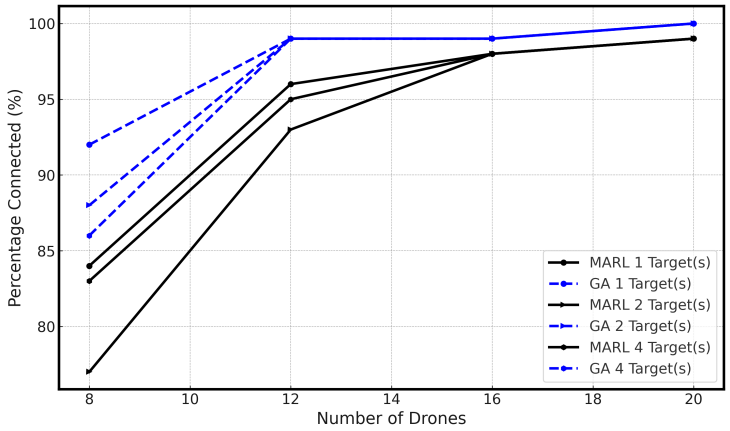
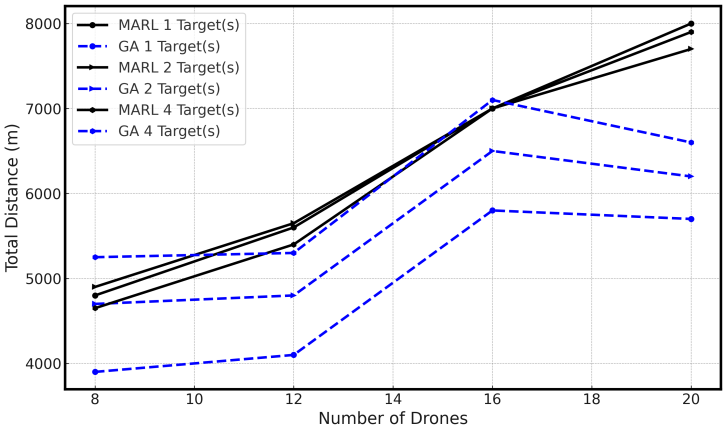


Figure 4.11: *Total travelled distances vs N_d.*



of the drones, GCS, and targets. With the usage of CNN and DQN, agents could make optimal joint connectivity and sensing decisions at any state provided that the search space is explored at a sufficient rate. The evolutionary algorithm based implementation has aimed to generate prior mission plans with high connectivity rates. Even though high connectivity rates could increase the probability of immediate informing from a randomly located target to the GCS, if upon its detection, the detecting drone is disconnected, a dynamic reconfiguration is required to form a relay chain while continuing the search for new targets. Our future work will focus on inclusion of limitations on sensing and heterogeneous targets into the path planner model.

5. ADAPTIVE MULTI-UAV COORDINATION FOR HETEROGENEOUS TARGET SEARCH AND CONNECT MISSIONS USING PROXIMAL POLICY OPTIMIZATION

5.1 Introduction

Until this chapter, we have investigated path planners based on traditional and RL-based optimizers. Although we have achieved a set of acceptable solutions for many different cases, our models have focused on a single mission completion criterion with a homogeneous target distribution. In this chapter, we consider an SAR mission, where a multi-UAV team is tasked with searching a disaster-stricken area to detect multiple targets with different information update priorities while maintaining connectivity to a GCS.

We *propose a novel MARL model to detect and connect multiple heterogeneous targets in dynamic multi-UAV SAR missions*. Our multi-UAV mission involves two phases: i) *Target search phase*, where UAVs explore the disaster area to locate targets, and ii) *target information phase*, where UAVs form a multi-hop ad hoc wireless network to transmit target data to the base station with different update requirements. We consider three target types [60]: i) targets that require connectivity for successive time intervals (e.g., disaster victims); ii) targets that require multiple nonconsecutive updates (e.g., critical infrastructure, or evacuation paths); iii) targets that require one-time best-effort update. For this mission, we introduce a novel implementation of PPO tailored for multi-UAV coordination, which is *configurable* for various circumstances with different priorities given to the target detection and connection related tasks.

Compared to commonly used models such as DQN, Advantage Actor-Critic (A2C), or DDPG, PPO leads to greater mission success and promotes stable learning [61]. *To optimize mission operation, we propose a dynamic reward function that combines area*

exploration, UAV-GCS connectivity, target detection, target connectivity based on information priority to allow flexible mission designs with coverage and connectivity aspects. This joint optimization approach significantly improves methods that use fixed reward structures or optimize rewards and learning parameters separately.

While we train our model for a fixed target type and a fixed grid size, we test the model for different target configurations and area sizes, illustrating its adaptability to different environments. Furthermore, we show the performance for coverage and connectivity tasks by setting the appropriate weights in our dynamic reward function and illustrate its use for missions with different goals. The results are consistent with classical multi-UAV path planning solutions; i.e., the model is able to capture the essence of optimal UAV paths efficiently.

The main contributions of this chapter are as follows:

- **Heterogeneous Target Framework:** A novel MARL architecture is presented. Building upon the previous MARL framework, the advanced version effectively handles multiple target types with varying information update requirements.
- **Dynamic Reward Integration:** A multi-objective reward function that adaptively balances exploration, connectivity, and target-specific requirements is introduced.
- **Quantitative Validation:** The proposed model shows consistent performance with classical optimization solutions while providing enhanced adaptability to dynamic mission requirements.

The chapter is organized as follows: Section 5.2 provides the environment, UAV, and target models. Section 5.3 explains the RL environment and the reward mechanisms. Section 5.4 explains our PPO model and training process. Section 5.5 showcases the performance of the model. Finally, Section 5.6 summarizes the chapter.

5.2 System Model

5.2.1 Environment and UAV model

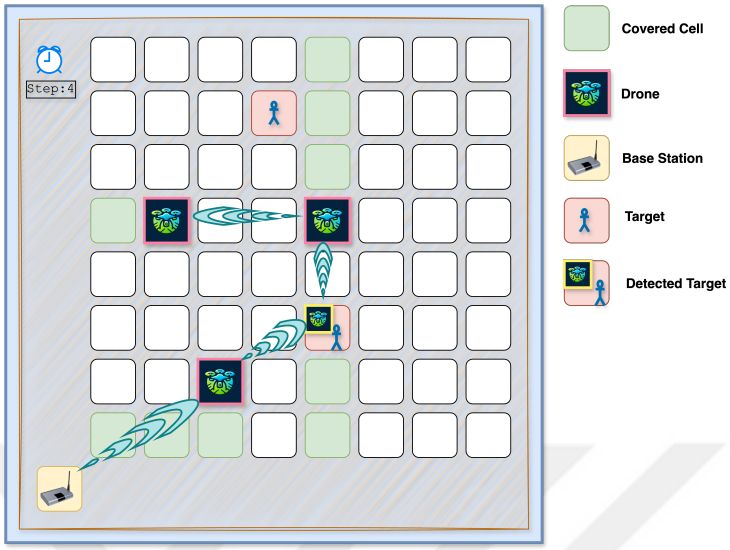
Our system simulates urban SAR missions after a disaster utilizing multiple UAVs (see Figure 5.1). UAVs search a potentially destroyed city for survivors and damaged infrastructure and report to a central command center. We represent the urban environment as a 2D grid and assume a constant flying altitude without the presence of obstacles. The grid is split into equal-sized cells, where cell size is determined by the ground coverage of onboard sensors. The paths of the UAVs correspond to a sequence of cells that will be visited at consecutive time steps. UAVs can efficiently explore cities with high maneuverability and omnidirectional movement. UAVs can adjust their speed to keep in sync, ensuring coordinated movement and continuous time steps throughout the swarm. A maximum time-step limit in our model represents UAV battery life. The UAVs are recharged and updated at the base station between missions. UAV team forms a dynamic and mobile ad-hoc network. The communication range is modeled as a disc based on our experimental results, where UAVs are equipped with a multi-antenna setup that provides omnidirectional radiation [43].

5.2.2 Perception and Target Modeling

We consider three types of static targets, each with distinct information requirements (Figure 5.2) based on their priorities as defined in our previous work [60].

1. Type 1: Requires consecutive information flow, e.g., disaster victims (high priority).
2. Type 2: Requires initial confirmation and multiple nonconsecutive updates, e.g., critical infrastructure, roads, damaged structures (medium priority).
3. Type 3: Requires one-time best-effort reporting (low priority).

Figure 5.1: *Sample mission diagram.*



The cameras and image processing on board allow each UAV to recognize and classify the various targets within their cell. We assume perfect sensing in each cell; i.e., target detection happens when a target and at least one UAV are in the same cell. Uncertainties due to environmental considerations and sensor constraints will be added to our sensing models in our future work.

5.3 RL Environment and Agents

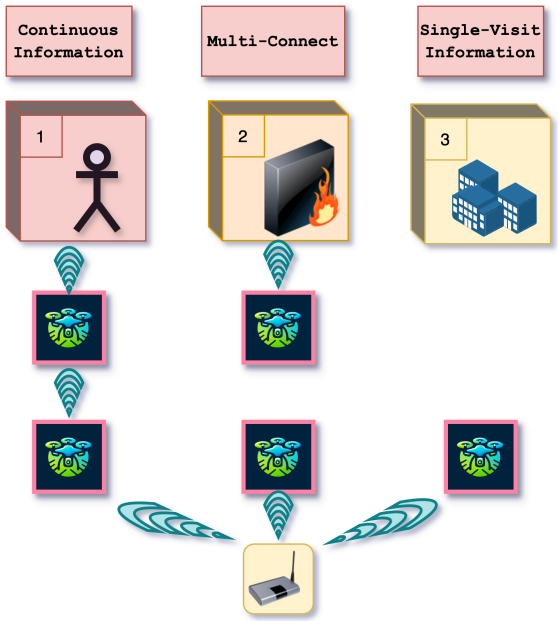
RL is often used for dynamic decision-making. An RL model has agents and an environment, where environmental influences reward or punish agents, causing observations. These observations tell the AI how to perform best in diverse situations. The proposed system allows users to change grid size, agent count, target categories, and communication range.

5.3.1 State Representation

The state s_t at time t is represented as:

$$s_t = [G_t, P_t, A_t, T_t] \tag{5.1}$$

Figure 5.2: *Target Types.*



where G_t is the normalized grid status corresponding to how many times a cell is visited, P_t are the normalized agent positions, A_t is the adjacency matrix representing connectivity, and T_t is the target detection state.

Rather than being explicitly assigned, UAV roles emerge naturally from the learned policy as they interact with the environment. All information collected by UAVs is sent to a ground control station located at a reference point (0,0), creating a centralized information flow. The location data transmitted by UAVs is compact, containing position, time, and target type information.

5.3.2 Action Space

Each agent has a discrete action space A = {up, down, left, right, up-right, up-left, down-right, down-left, hover}, representing possible movements. At each time step, an agent can move in the mentioned directions or stay in its current cell, depending on the detection and connectivity status.

5.3.3 Reward Mechanism Design

The effectiveness of multi-UAV RL models heavily rely on a carefully crafted reward mechanism.

Coverage, task performance, target recognition, and time consumption are examined in [62], where adaptive relay systems, connection maintenance, or relay networks are not considered. Coverage, connection, and work completion are prioritized over adaptive reward systems, target identification, and relay chains in .

In Chapter 4, we considered parameters such as coverage, connection, work completion, relay chain design, target identification, and time efficiency, where an adaptive reward system and subtask motivation are absent. Without adaptive multi-phase reward mechanisms, decision-making becomes more reliant on specific action selection biased towards environment context, being far away from adaptability. In this work, we propose a novel multi-phased reward structure that encourages behaviors aligned with the mission objectives while discouraging counterproductive actions to boost success in dynamic situations and provides a reliable UAV coordination technique. UAV collaboration is facilitated via coverage, connectivity, adaptive incentive systems, and subtask rewards.

The reward mechanism comprises several components, each addressing a specific aspect of the mission:

- **Exploration Reward** (R_{exp}): Encourages UAVs to cover unexplored areas, promoting efficient search patterns.
- **Discovery Penalty** (R_{disc}): Penalizes revisiting already explored cells, pushing UAVs to prioritize unexplored regions.
- **Connectivity Reward** (R_{conn}): Rewards UAVs for maintaining a connection to the GCS, either directly or through other UAVs.

- **Target Detection Reward (R_{det})**: A one-time reward given when a UAV detects a new target.
- **Target Information Relay Reward (R_{relay})**: Awarded when a UAV successfully relays information about a detected target back to the GCS.
- **Distance To Target Reward (R_{dis})**: This reward encourages UAVs to remain in proximity to detected targets.
- **Mission Completion Reward (R_{comp})**: A substantial reward given upon successful completion of all mission objectives.

These rewards are combined into the following function:

$$\begin{aligned} R_{total} =& w_1 R_{exp} + w_2 R_{disc} + w_3 R_{conn} \\ &+ w_4 R_{det} + w_5 R_{relay} + w_6 R_{dis} + w_7 R_{comp} \end{aligned} \tag{5.2}$$

where w_i's are weight parameters that can be tuned to prioritize different aspects of the mission.

This reward structure allows for flexible mission designs. By adjusting the weights, we can emphasize different aspects of the mission. For example, increasing w_3 would prioritize maintaining connectivity, which might be crucial in scenarios where real-time information exchange is critical. Conversely, a higher w_1 would encourage more aggressive exploration, suitable for time-critical search operations.

In this chapter, we consider two main mission phases:

1. **Target Search Phase**

   UAVs are released to the arena from specified start points. They look for targets in the search grid, while maintaining connectivity. The reward structure allows drones to choose optimal paths for coverage, while maximizing connectivity. After all the targets are detected, the mission switches to target information phase.

2. **Target Information Phase** Drones are rewarded if they form relay chains from the target detecting drone to the GCS. Each step is evaluated on the average reward per drone meaning that teamwork is awarded. Ideally, in such an SAR mission, we desire a UAV team configuration such that UAVs spread in the search area as fast as possible, while staying connected to the GCS. This might lead to some UAVs hovering in their position to assist connectivity, whereas others moving out to find targets.

5.4 PPO for Multi-UAV Coordination

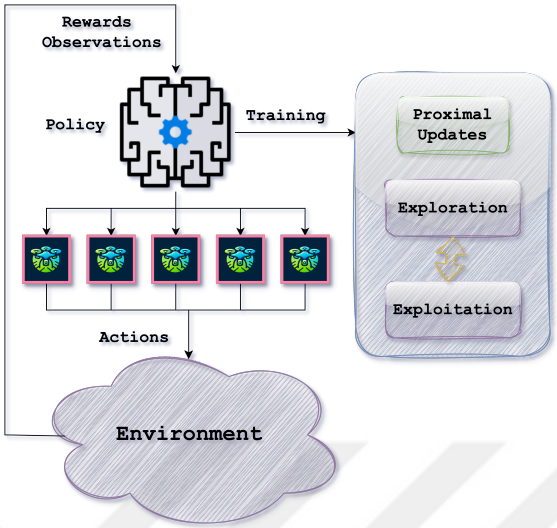
After extensive experimentation with various RL approaches, we observed that PPO consistently outperformed other methods, including DDPG [63], and DQN (Chapter 4) in solving our complex multi-UAV coordination problem. Table 5.1 shows a qualitative comparison between PPO, DDPG, and DQN.

Table 5.1: *Comparison of PPO, DDPG, and DQN for Adaptive Strategy Missions.*

Criteria	**PPO**	**DDPG**	**DQN**
Adaptability	High	Moderate	Low
Stability	High	Moderate	Low
Exploration	Balanced	Low	High
Sample Efficiency	Moderate	High	Low
Training Time	Moderate	Long	Short

In this work, we develop a PPO based method (see Figure 5.3). We introduce a dynamic reward adjustment taking form within the hyperparameter tuning process without affecting the final model's complexity, only increasing its effectiveness. Then we split the mission into different phases where specific parameters determining long-short term strategies are set according to mission requirements. For instance, coverage is a task requiring correct actions decided on correct time instances, a model does not need to consider a path it will take at the end of the mission when determining its current path,

Figure 5.3: *PPO overview.*



as the entire mission is of equal importance: covering new cells. A target inform phase is different, there is a clear goal where a target information must be delivered to GCS by forming a relay chain from the target to the GCS.

5.4.1 Training Procedure

Our training procedure follows the PPO algorithm for multi-agent scenarios. The key steps are given in Algorithm 4. The procedure alternates between data collection and policy optimization. The clipped objective function enforces stable learning by limiting policy updates at each step, while the value function update improves state value estimation. The search and inform phases are trained separately with their own reward functions. The training is done for multiple type 2 targets; but we have verified that the same model can be used for different target types. This is a *major benefit* since the model can be used *for different target configurations without the need for re-training*. Training is conducted over a million different randomly distributed targets.

Algorithm 4 PPO Training Procedure.

Initialize environment with parameters: grid size, number of agents, number of targets, communication range.
Initialize PPO agent with hyperparameters including learning rate, discount factor (γ), entropy coefficient, value function coefficient, number of epochs, batch size
for episode $\leftarrow 1$ to total_episodes **do**
    $state \leftarrow$ reset_environment()
    $done \leftarrow$ False
    $timestep \leftarrow 0$
    while not done **do**
        action $\leftarrow$ agent.select_action(*state, exploration_rate*)
        new_state, reward, done, info $\leftarrow$ take_action($action$)
        agent.store_experience(*state, action, reward, new_state, done*)
        state $\leftarrow$ new_state
        $timestep \leftarrow timestep + 1$
        if $timestep \bmod update\_interval == 0$ **then**
            agent.update_policy(ϵ, *learning_rate,* γ)
        end if
    end while
    exploration_rate $\leftarrow$ max(*min_exploration_rate*,
*exploration_rate * exploration_decay_rate)*
end for

5.4.2 Model Parameters

We use Optuna, a hyperparameter optimization library for optimizing model parameters. We explore each critical parameter with their mathematical importance to the training procedure.

1. **Reward Values** Each reward value is optimized considering the trained policy's mission phase. Some reward functions are omitted after optimization, such as distance to target reward which is the most frequent reward in the second phase. Therefore, while in theory a proposed reward function may work in favor of the model, in reality, it can hinder the training process. It can also be observed that limiting the frequency of rewards can strengthen long term planning.

2. **Discount Factor (γ)** This value is optimized individually for different phases. It

determines the amount of short-term and long-term planning of the agents. Agents need to make short-term decisions with frequent rewarding on discoveries requiring a lower discount factor. On the other hand, they need to make long term decisions while forming relay chains which requires a larger γ value.

Other parameters such as clip range (set to 0.2) and entropy coefficient which determines the exploration rate are also optimized. These parameters have an optimal range of values on both phases given in the next section.

5.4.3 Scalability Analysis

The computational complexity of our model scales as $O(N^2 + mn|T|)$ per time step, where N is the number of UAVs, $m \times n$ is the grid size, and $|T|$ is the number of target types. The quadratic term N^2 comes from updating the adjacency matrix, while $mn|T|$ represents the cost of updating the grid and target information.

5.5 Results and Analysis

The proposed multi agent PPO model is evaluated on various performance criteria based on grid size, number of drones, and target configurations.

5.5.1 Experimental Setup and Parameters

The grid size is increased from 8x8 to 16x16. The number of drones is changed from 5 to 15, with transmission range set to $2\sqrt{2}$ cells. This range and number of drones combination lead to highly successful missions for small areas; however, for larger areas the spatial density is not sufficient to accomplish all the connect and inform tasks as will be shown. Targets are randomly distributed in the grid area in the following configurations that represent the versatility of our model.

- [2]: Single target of type 2 (detected targets need to be connected to the GCS opportunistically at least 3 times)

- $[2, 2]$: Two targets of type 2
- $[2, 2, 2]$: Three targets of type 2
- $[1, 2, 3]$: One target of each type (heterogeneous scenario, where type 1 target needs to be connected to the GCS for 3 consecutive timesteps)

The PPO algorithm was implemented with the following key parameters: Learning rate is set to 0.00003 for stable learning. Discount factor (γ) is 0.78 for the first phase and 0.88 for the second phase to balance short-term and long-term rewards. Entropy coefficient is 0.01 to encourage exploration, whereas value function coefficient is 0.5 to balance policy and value function updates. Number of epochs is 10 for each policy update and batch size is 1024 for stable gradient estimates. The rewards for target search and inform phases are given in Table 5.2. These PPO parameter values are chosen after extensive analysis for optimal performance.

Table 5.2: *Reward values for Target Search and Target Inform phases.*

Reward	R_{exp}	R_{disc}	R_{conn}	R_{det}	R_{relay}	R_{dis}	R_{comp}
Search Phase	10	-5	5	-	-	-	-
Inform Phase	-	-	-	7	14	0	5

5.5.2 Training Setup

The model is trained on a single Intel Core i7 9750h Central Processing Unit (CPU), indicating its simple but effective nature as it can be trained on almost any configuration. Each training is run for 1.5 million timesteps, which approximately took 45 minutes to complete. Inference time for each time step through a mission is around 1 second. The training is done for type 2 targets only, while tests are done for both homogeneous and heterogeneous target configurations.

5.5.3 Performance Parameters

- **Coverage time**: Time taken from take-off till whole grid is covered.
- **Average Inform Time**: Time taken from visiting the targets to informing the GCS by forming relay chains, given the detected target's update requirements.
- **Percentage Connectivity**: Overall percentage connected time to GCS at target inform phase.
- **Average number of covered cells**: Number of cells that are visited at least once during the target inform phase. Target search phase covers the entire grid.

5.5.4 Coverage times

First, we validate our model for a coverage only mission; i.e., the UAV team explores a given area which corresponds to the target search phase such that coverage time (from take-off to landing) is minimized. Table 5.3 shows the coverage times for different drone numbers for an 8x8 grid, where the results are averaged across multiple iterations. As a comparison, we also provide the coverage time for a multi-UAV path plan obtained using GA that minimizes coverage time presented in Chapter 3. The PPO results are consistent with the optimal GA results.

Table 5.3: *Coverage Times in s for Different Numbers of Drones.*

Drones	5	7	9	11	13	15
PPO	618	578	538	496	466	448
GA	592	557	521	484	459	446

5.5.5 Inform times

Next, we investigate the impact of various parameters on the average inform times. Our model is trained for 8x8 grid and tested on different area sizes. Note that depending

on the flight time mission might not be completed, especially when the grid size is too large and/or the transmission range is too low, i.e., a relay chain cannot be formed from all targets. We set the limit for such targets to 800s in the figures to avoid biasing results toward successful completions. Figure 5.4 shows the average number of completed target connection tasks for various system parameters when all targets are of type 2. As expected, the likelihood that all target inform tasks are accomplished, i.e., each target information is updated three times, is higher when the area size is small. As the area and the number of targets grow, the successful inform tasks might drop to 60%. Figure 5.5 illustrates the relationship between grid size, number of targets, and average inform time for 15 drones and type 2 targets. As the grid size and target count increase, the inform time also increases as expected since the targets are distributed over a larger area and formation of relay chains to inform the GCS becomes more time-consuming. The rate of increase in inform time reduces as the area size increases for increasing number of targets.

Figure 5.4: *Average number of completed targets for different numbers of targets.*

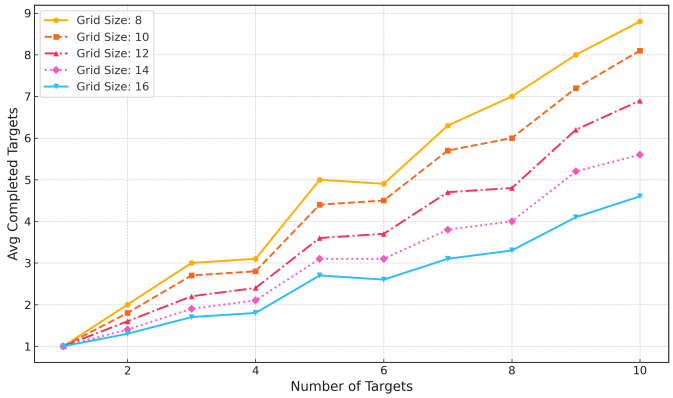


Figure 5.6 compares inform times for different target configurations across various grid sizes for 15 drones. The [1, 2, 3] configuration (representing heterogeneous targets) consistently requires the longest inform times across all grid sizes, suggesting that dealing with diverse target types poses a greater challenge to the system. Conversely, the [2]

Figure 5.5: *Average Inform Time vs Number of Targets for Different Grid Sizes.*

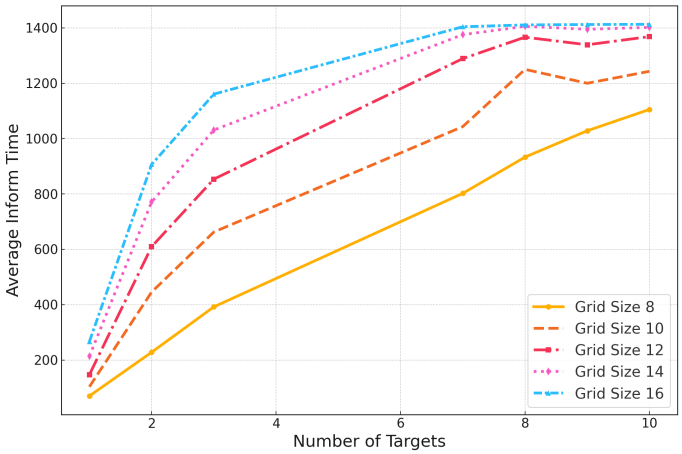
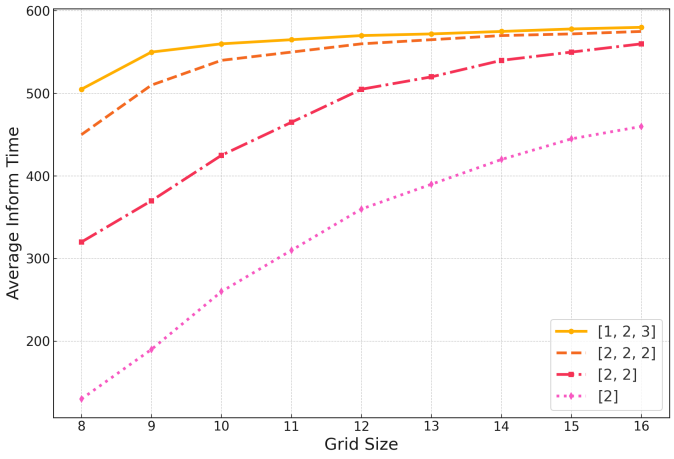


Figure 5.6: *Average Inform Time vs Grid Size for Different Target Configurations.*



configuration (single target) results in the shortest times since drones need to focus on a single target. As the grid size increases, the changes become more apparent since the models showcase their effectiveness in larger environments with more ease. Figure 5.7 shows how the number of drones affects inform times across different grid sizes for the [1,2,3] target configuration (the worst case). Observe that for smaller grids increasing the

Figure 5.7: *Average Inform Time vs Number of Drones for Different Grid Sizes.*

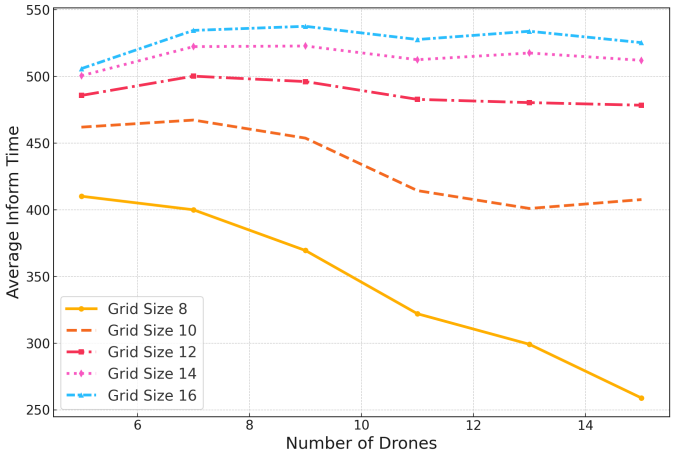
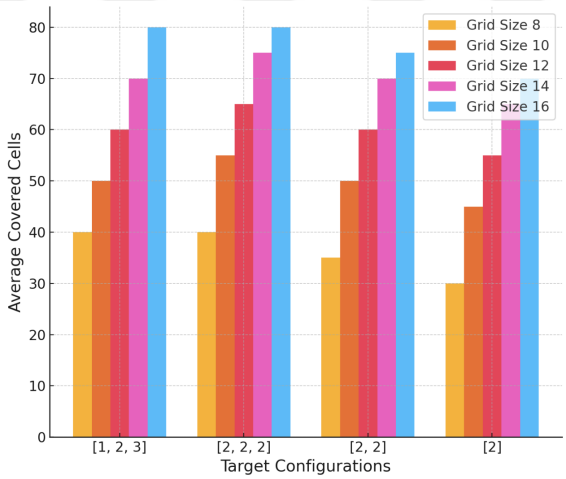
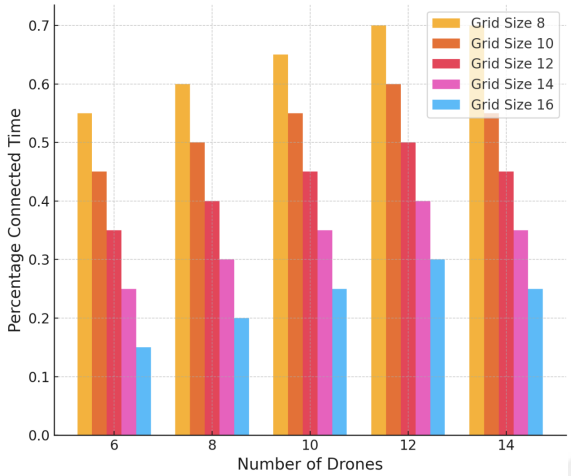


Figure 5.8: *Covered Cells vs Target Configurations.*



number of drones decreases the inform time since there is enough redundancy in terms of forming connected graphs to GCS. As the grid size increases, the drones spread out in the search area and likelihood of finishing the mission successfully reduces especially for small number of drones, i.e., a minimum number of drones is necessary to guarantee relay chain formation from the targets to the GCS. The inform time converges for larger grid sizes since the number of satisfied target requirements reduce (as shown in Figure 5.6).

Figure 5.9: *Percentage Connected Time vs Number of Drones.*



5.5.6 Area Coverage and Connectivity

Figure 5.8 shows the effectiveness of area coverage during the inform phase for different target configurations and grid sizes. The results are averaged over number of drones changing from 5 to 15. Covered cells consistently increase with grid size for all configurations showing that drones spend more time on the search area covering cells and forming relay chains. The [2, 2, 2] configuration (three targets of the same type) achieves higher coverage across most grid sizes, suggesting that multiple targets scattered around the search area result in drones dividing and covering a wide area in a uniform manner compared to the [1,2,3] configuration.

Figure 5.9 illustrates how connectivity is maintained across different grid sizes and numbers of drones for [1,2,3] target configuration. Connectivity generally decreases as grid size increases, which is expected due to the larger area that needs to be covered. For most grid sizes, connectivity improves as the number of drones increases and converges. This suggests an optimal range can be found for the number of drones in maintaining network connectivity, beyond which additional drones may not provide significant benefits

and could potentially interfere with efficient communication patterns. This can be explained by the multi-agent nature of the problem where coordination is easier with more compact models.

5.6 Summary

In this chapter, a PPO-based multi-UAV coordination framework is proposed for missions with changing requirements such as SAR. The model considers area exploration and UAV connectivity, as well as target detection and connection, where the targets have different information update requirements. We propose a dynamic reward function that captures these aspects and utilize the function for different phases of a multi-target SAR mission. While the model is trained for a fixed target configuration and area size, we test its performance for a range of different scenarios and illustrate the adaptability and versatility, and hence, reusability of our model for missions with possibly different task priorities than our own.

6. CONCLUSIONS

This thesis has presented a systematic advancement of multi-UAV path planning techniques, from classical optimization strategies to state-of-the-art solutions based on deep reinforcement learning. Our principal aim has been to address two key demands of operational multi-UAV scenarios: The need for real-time adaptability and the need to balance competing objectives.

The key contributions and findings of this thesis can be summarized as:

- **Novel Multi-Objective Framework:** We developed a multi objective optimization framework for conflicting objectives such as area coverage time and percentage connected time to GCS. Overall, our model succesfully generated a pareto front of waypoint based UAV paths.
- **Dynamic Decision Making:** Building on the static optimization framework, we introduced a DQN-based reinforcement learning approach that improved target detection times by 20% while maintaining connectivity above 85%. This demonstrated the potential of learning-based approaches for real-time adaptation.
- **Adaptive Strategy Selection:** Based on our DQN-based model, we developed a more sophisticated PPO-based partially observable markov state space model that can adapt to different types of targets and mission phases.

Several promising directions for future research emerge from this work. Integration of sensing uncertainties and environmental constraints into the learning models is our prior concern for further applicability in real-life scenarios. Another goal is the extension of the framework to handle dynamic targets with probabilistic state changes.

Progress in the path planning of multiple autonomous drones shows how well artificial intelligence can work when it comes to dynamic mission requirements. This

progress was made possible by using part of the foundations of deep learning and the tools that it provides. Our focus was on the operation of multiple drones during search and rescue missions. But the tools and algorithms that we used have potential application in many different areas, especially those that demand real-time operation and decision making by coordinated multiple agents, such as community planning and environmental monitoring.

REFERENCES

[1] M. Mozaffari, W. Saad, M. Bennis, Y.-H. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2334–2360, 2019.

[2] S. Hayat, E. Yanmaz, and R. Muzaffar, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Communications Surveys Tutorials*, vol. 18, pp. 2624–2661, Fourthquarter 2016.

[3] A. Otto, N. Agatz, J. Campbell, B. Golden, and E. Pesch, "Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey," *Networks*, vol. 72, no. 4, pp. 411–458, 2018.

[4] C. Zhang, P. Patras, and H. Haddadi, "Deep reinforcement learning for UAV navigation through massive mimo technique," *IEEE Transactions on Mobile Computing*, vol. 18, no. 8, pp. 1838–1850, 2019.

[5] E. Yanmaz, "Connectivity considerations for mission planning of multiple UAVs," *Turkish Journal of Electrical Engineering and Computer Sciences*, vol. 28, pp. 2228–2243, 2020.

[6] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, "Help from the sky: Leveraging UAVs for disaster management," *IEEE Pervasive Computing*, vol. 16, no. 1, pp. 24–32, 2017.

[7] N. Zhao, W. Lu, M. Sheng, Y. Chen, J. Tang, F. R. Yu, and K.-K. Wong, "UAV-assisted emergency networks in disasters," *IEEE Wireless Communications*, vol. 26, no. 1, pp. 45–51, 2019.

[8] E. J. Glantz, F. E. Ritter, D. Gilbreath, S. J. Stager, A. Anton, and R. Emani, "UAV use in disaster management," *International Conference on Information Systems for Crisis Response and Management*, 2020.

[9] F. Greenwood, E. L. Nelson, and P. G. Greenough, "Flying into the hurricane: A case study of UAV use in damage assessment during the 2017 hurricanes in Texas and Florida," *PloS one*, vol. 15, no. 2, p. e0227808, 2020.

[10] M. A. Ruiz Estrada and I. Ndoma, "The uses of unmanned aerial vehicles–UAV's–(or drones) in social logistic: Natural disasters response and humanitarian relief aid," *Procedia computer science*, vol. 149, pp. 375–383, 2019.

[11] R. Majumder, S. Jana, P. P. Menon, D. Ghose, N. Prusty, B. Mukherjee, and A. Ghosh, "Cyclone preparedness, rescue operations and damage assessment using UAV," *arXiv preprint arXiv:2203.06497*, 2022.

[12] T. Manzini, R. Murphy, and D. Merrick, "Quantitative data analysis: Crasar small unmanned aerial systems at hurricane ian," *arXiv preprint arXiv:2308.14577*, 2023.

[13] İ. Güven and E. Yanmaz, "Multi-objective path planning for multi-uav connectivity and area coverage," *Ad Hoc Networks*, vol. 160, 2024.

[14] İ. Güven and E. Yanmaz, "Maintaining connectivity for multi-uav multi-target search using reinforcement learning," in *Proc. of the Int'l ACM Symp. Design and Analysis of Intel. Vehic. Netw. and Appl.*, 2023.

[15] R. Bellman, "On a routing problem," *Quarterly of Applied Mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[16] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[17] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[18] L. E. Parker, "Alliance: An architecture for fault tolerant multi-robot cooperation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 2, pp. 220–240, 1998.

[19] A. G. Barto, S. J. Bradtke, and S. P. Singh, "Learning to act using real-time dynamic programming," *Artificial Intelligence*, vol. 72, no. 1-2, pp. 81–138, 1995.

[20] F. K. Dhruv Karve, "Multi-UAV path planning using modified dijkstra's algorithm," *International Journal of Computer Applications*, vol. 175, pp. 26–33, Oct 2020.

[21] Z. Ren, S. Rathinam, and H. Choset, "Multi-objective conflict-based search for multi-agent path finding," *CoRR*, vol. abs/2101.03805, 2021.

[22] A. Puente-Castro, E. Fernandez-Blanco, and D. Rivero, "Genetic algorithm based system for path planning with unmanned aerial vehicles swarms in cell-grid environments," 2024.

[23] C. Huang and J. Fei, "UAV path planning based on particle swarm optimization with global best path competition," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 32, 10 2017.

[24] Z. Ren, S. Rathinam, and H. Choset, "Multi-objective conflict-based search using safe-interval path planning," *CoRR*, vol. abs/2108.00745, 2021.

[25] M. Singh, A. Choudhary, S. Gulia, and A. Verma, "Multi-objective NSGA-II optimization framework for UAV path planning in an UAV-assisted WSN," *The Journal of Supercomputing*, vol. 79, pp. 1–35, 07 2022.

[26] F. Rekabi Bana, T. Krajník, and F. Arvin, "Evolutionary optimization for risk-aware heterogeneous multi-agent path planning in uncertain environments," *Frontiers in robotics and AI*, vol. 11, p. 1375393, 08 2024.

[27] A. Pohl and G. Lamont, "Multi-objective UAV mission planning using evolutionary computation," *2008 Winter Simulation Conference*, pp. 1268–1279, 12 2008.

[28] A. Mardani, M. Chiaberge, and P. Giaccone, "Communication-aware UAV path planning," *IEEE Access*, vol. 7, pp. 52609–52621, 01 2019.

[29] A. Viseras, Z. Xu, and L. Merino, "Distributed multi-robot cooperation for information gathering under communication constraints," *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1267–1272, 2018.

[30] J. Sánchez-García, D. Gutiérrez, and S. Toral, "A distributed PSO-based exploration algorithm for a UAV network assisting a disaster scenario," *Future Generation Computer Systems*, vol. 90, 07 2018.

[31] Y. Jiang, "Optimizing risk-aware task migration algorithm among multiplex UAV groups through hybrid attention multi-agent reinforcement learning," *Tsinghua Science & Technology*, 2025.

[32] H. Qie, D. Shi, T. Shen, X. Xu, Y. Li, and L. Wang, "Joint optimization of multi-UAV target assignment and path planning based on multi-agent reinforcement learning," *Ieee Access*, 2019.

[33] Z. Jiang, Y. Chen, G. Song, B. Yang, and X. Jiang, "Cooperative planning of multi-UAV logistics delivery by multi-graph reinforcement learning," in *International Conference on Computer Application and Information Security (ICCAIS 2022)* (V. Varadarajan, J. C.-W. Lin, and P. Lorenz, eds.), vol. 12609, p. 126090I, International Society for Optics and Photonics, SPIE, 2023.

[34] B. Omoniwa, B. Galkin, and I. Dusparić, "Optimizing energy efficiency in UAV-assisted networks using deep reinforcement learning," *Ieee Wireless Communications Letters*, 2022.

[35] D. Zheng, C. Liu, and L. Huang, "Spatio-temporal coverage planning algorithm for multi-UAV and multi-sensor cooperative search," *Journal of Physics Conference Series*, 2022.

[36] M. Yang, G. Liu, Z. Zhou, and J. Wang, "Partially observable mean field multi-agent reinforcement learning based on graph attention network for UAV swarms," *Drones*, 2023.

[37] M. N. Ndiaye, E. H. Bergou, and H. E. Hammouti, "Muti-agent proximal policy optimization for data freshness in uav-assisted networks," 2023.

[38] D. Wang, "Three-dimensional trajectory and resource allocation optimization in multi-unmanned aerial vehicle multicast system: A multi-agent reinforcement learning method," *Drones*, 2023.

[39] M. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: fundamentals and evolutionary methods," *Natural Computing*, vol. 17, pp. 585 – 609, 2018.

[40] J. Blank and K. Deb, "Pymoo: Multi-Objective optimization in python," *IEEE Access*, vol. 8, pp. 89497–89509, 2020.

[41] A. Khan, E. Yanmaz, and B. Rinner, "Information Exchange and Decision Making in Micro Aerial Vehicle Networks for Cooperative Search," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 4, pp. 335–347, 2015.

[42] E. Yanmaz, "Connectivity considerations for mission planning of a search and rescue drone team," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 28, pp. 2228–2243, 2020.

[43] E. Yanmaz, R. Kuschnig, and C. Bettstetter, "Achieving air-ground communications in 802.11 networks with three-dimensional aerial mobility," in *Proc. IEEE Int. Conf. on Computer Communications (INFOCOM), Mini conference*, Apr. 2013.

[44] A. Molaei and E. Yanmaz, "Network analysis of connectivity optimized multi-UAV path planners," in *Proc. IEEE Conf. Comp. Commun. Workshops (INFOCOM WKSHPS)*, May 2024.

[45] K. Deb, "Multi-objective optimization using evolutionary algorithms," in *Wiley-Interscience series in systems and optimization*, 2001.

[46] A. Osyczka, "An approach to multicriterion optimization problems for engineering design," *Computer Methods in Applied Mechanics and Engineering*, vol. 15, pp. 309–333, 1978.

[47] J.-H. Cho, Y. Wang, I.-R. Chen, K. S. Chan, and A. Swami, "A survey on modeling and optimizing multi-objective systems," *IEEE Communications Surveys & Tutorials*, vol. 19, pp. 1867–1901, 2017.

[48] M. Ehrgott and M. M. Wiecek, *Multiobjective Programming*, pp. 667–708. New York, NY: Springer New York, 2005.

[49] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Communications Surveys & Tutorials*, vol. 19, pp. 550–586, 2016.

[50] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, 2002.

[51] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, 2014.

[52] X. Yu and Mitsuo, *Introduction to evolutionary algorithms*. London, England: Springer, 2010 ed., 2012.

[53] D. Partridge, "A review of: "handbook of genetic algorithms" l. davis (ed.) new york: Van nostrand reinhold, 1991 ISBN 0-442-00173-8, 385pp., £32.50," *Conn. Sci.*, vol. 3, no. 4, pp. 446–448, 1991.

[54] Y.-C. Choi and H.-S. Ahn, "A survey on multi-agent reinforcement learning: Coordination problems," in *Proceedings of 2010 IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications*, IEEE, 2010.

[55] X. Wang, S. Wang, X. Liang, D. Zhao, J. Huang, X. Xu, B. Dai, and Q. Miao, "Deep reinforcement learning: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. PP, 2022.

[56] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, *PyTorch: an imperative style, high-performance deep learning library*. Red Hook, NY, USA: Curran Associates Inc., 2019.

[57] P. N. Shinde, "Pygame: Develop games using python," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. VI, pp. 4442–4447, 2021.

[58] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. Del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, "Array programming with NumPy," *Nature*, vol. 585, no. 7825, pp. 357–362, 2020.

[59] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Comput. Sci. Eng.*, vol. 9, no. 3, pp. 90–95, 2007.

[60] H. M. Balanji and E. Yanmaz, "Priority-based dynamic multi-UAV positioning for multi-target search and connectivity," in *IEEE Wireless Commun. and Netw. Conf. (WCNC)*, pp. 1–6, 2024.

[61] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv:1707.06347*, 2017.

[62] G. Ding, J. J. Koh, K. Merckaert, B. Vanderborght, M. M. Nicotra, C. Heckman, A. Roncone, and L. Chen, "Distributed reinforcement learning for cooperative multi-robot object manipulation," *CoRR*, vol. abs/2003.09540, 2020.

[63] D. Fan, H. Shen, and L. Dong, "Multi-agent distributed deep deterministic policy gradient for partially observable tracking," *Actuators*, vol. 10, no. 10, p. 268, 2021.