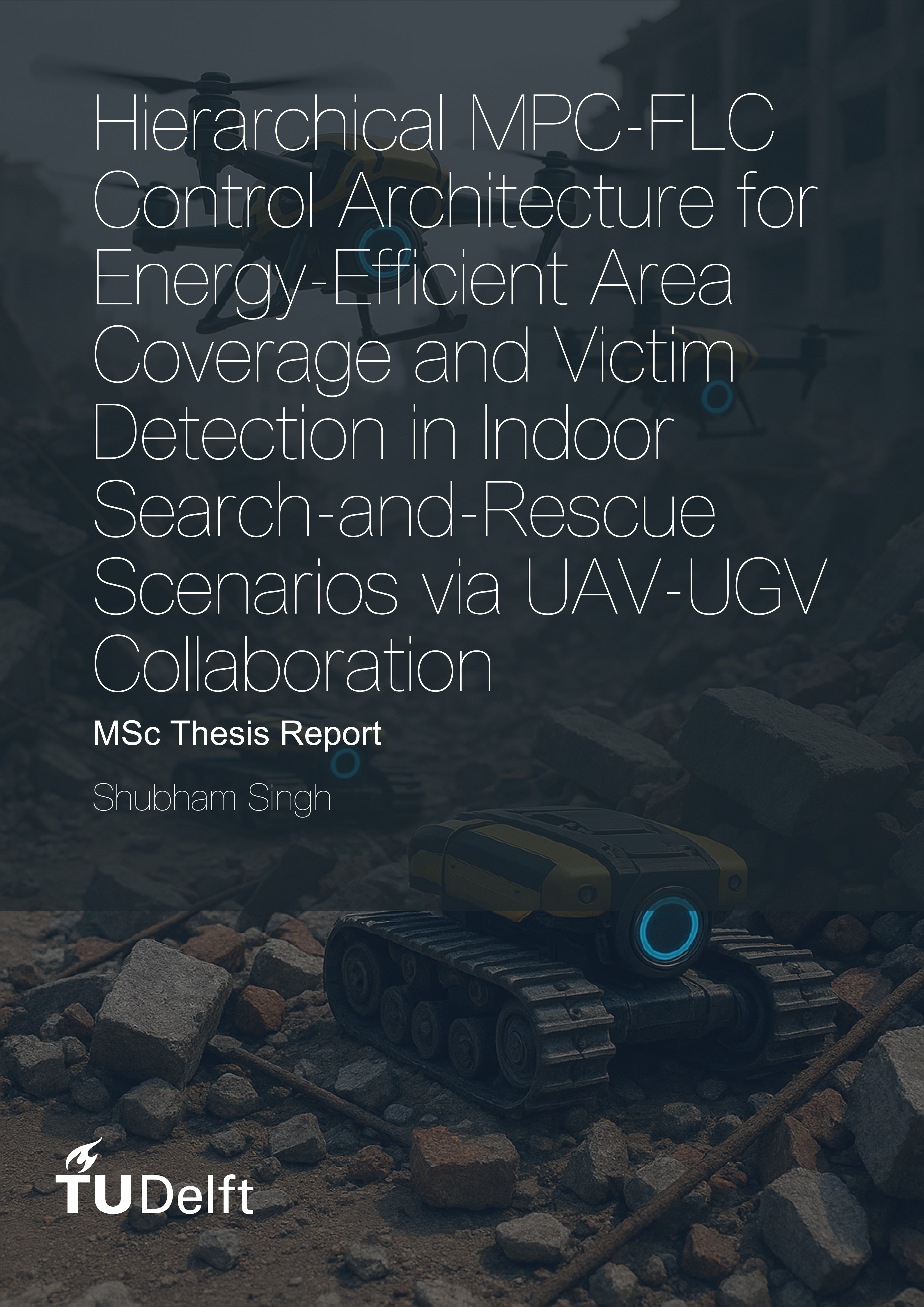


Hierarchical MPC-FLC Control Architecture for Energy-Efficient Area Coverage and Victim Detection in Indoor Search-and-Rescue Scenarios via UAV-UGV Collaboration

MSc Thesis Report

Shubham Singh

Hierarchical MPC-FLC Control Architecture for Energy-Efficient Area Coverage and Victim Detection in Indoor Search-and-Rescue Scenarios via UAV-UGV Collaboration

MSc Thesis report

by

Shubham Singh

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on May 6, 2025 at 13:00

Thesis committee:	
Chair:	Dr. MJ Ribeiro
Supervisor:	Dr. A Jamshidnejad
External examiner:	Dr.ir. E. Mooij
Place:	Faculty of Aerospace Engineering, Delft
Project Duration:	July, 2024 - May, 2025
Student number:	5694647

Faculty of Aerospace Engineering · Delft University of Technology

Preface

With this thesis, I conclude a rewarding and intense chapter of my academic journey at Delft University of Technology. In 2022, I moved from India to the Netherlands with the ambition of gaining a good engineering education and making my mark in the aerospace industry. This journey, though enriching, was filled with its fair share of ups and downs. There were moments when I was tested to what I thought were my limits. Particularly during the thesis phase, there were times when setbacks and challenges made me question whether I would be able to finish what I had started.

Yet, through it all, I was fortunate to be surrounded by people who believed in me. First and foremost, I would like to express my deepest gratitude to my supervisor, Dr. Anahita Jamshidnejad. Her steadfast support, rigorous feedback, and encouragement during the toughest parts of this research made all the difference. Beyond academic guidance, I have picked valuable lessons from her that I will carry forward in both my professional and personal life. For her patience, mentorship, and belief in my potential, I am sincerely grateful.

I am also deeply grateful to my friends (a.k.a chosen family): Shreya, Jagadish, Aashna, Kasana, Sayak, Gayathri, Komal, and Vidhi, who stood by me through the many highs and lows of this journey. Their support and reassurance gave me strength during the moments I needed it most.

Finally, and most importantly, I would like to thank my family. My brother, Krishnav, has been a constant source of inspiration and strength, even while embarking on his own important journey of starting his engineering education. Above all, I owe everything to my parents, whose unconditional love, sacrifices, and unwavering belief in me has got me to this point. The faith they have placed in me has been my guiding light, and it is my mission in life to give back to them all the love, comfort, and happiness they deserve.

-Shubham Singh

Contents

Nomenclature

List of Abbreviations

A*	A-star pathfinding algorithm
ACS	Ant Colony System
ASR	Air Sea Rescue
FBL	Feedback Linearization
FIS	Fuzzy Inference System
FLC	Fuzzy Logic Control
FMRLC	Fuzzy Model Reference Learning Control
GA	Genetic Algorithm
GNN	Graph Neural Network
LiDAR	Light Detection and Ranging
MPC	Model Predictive Control
MRTA	Multi-Robot Task Allocation
PID	Proportional–Integral–Derivative (control)
PSO	Particle Swarm Optimization
RL	Reinforcement Learning
SAR	Search-and-Rescue
SARS	Search-and-Rescue Systems
UAV	Unmanned Aerial Vehicle
UGV	Unmanned Ground Vehicle
UNDRR	United Nations Office for Disaster Risk Reduction
USAR	Urban Search-and-Rescue
VTOL	Vertical Take-Off and Landing
WiSAR	Wilderness Search-and-Rescue

Agent Characteristics

i	Agent index
j	Agent type (j=0 for UGV and j=1 for UAV)
$a_{i,j}$	represents the agent with index i and of type j ($a_{i,0}$ →UGVs and $a_{i,1}$ →UAVs where $i = 1 \ldots N$ and N is the total number of agents in the system
$(x_i,\ y_i)$	Coordinates of the cell in which agent is located at current time
$r^{\mathsf{p}}_{i,j}$	Perception radius of agent $a_{i,j}$
$E_i(x_i, y_i, r^{\mathsf{p}}_{i,j})$	Perception field of agent $a_{i,j}$ at (x_i, y_i) (all cells within $r^{\mathsf{p}}_{i,j}$)
$\eta_{i,j}$	Perceptual uncertainty reduction rate for agent $a_{i,j}$
$\alpha_{i,j}$	Occlusion sensitivity factor for agent $a_{i,j}$

Environment Parameters

E	Set of all cells in the environment
$(x,\ y)$	Coordinates of a cell in the grid environment
(L_x, L_y)	Discrete dimensions of environment
$(x^{\mathsf{V}}_v(\tau), y^{\mathsf{V}}_v(\tau))$	Position of victim v at time step τ
p_s	Probability of a victim v remaining in current cell
p^{m}_v	Probability of a victim v moving to a neighbouring cell
$n^{\mathsf{free}}_v(x^{\mathsf{V}}_v(\tau), y^{\mathsf{V}}_v(\tau), \tau)$	Number of free neighbouring cells for cell $(x^{\mathsf{V}}_v(\tau), y^{\mathsf{V}}_v(\tau))$
$t(x, y)$	Terrain difficulty index (0 = easiest, 1 = most difficult)
$o(x, y)$	Observability index (0 = poorest, 1 = highest)
$c(x, y, \tau)$	Scan certainty for cell (x, y) at time τ
$\mathcal{C}(\tau)$	Scan certainty map
$\mathcal{W}(\tau)$	Occupancy map
$\mathcal{V}(\tau)$	Victim map
$\mathcal{T}(\tau)$	Terrain Map
$\mathcal{O}(\tau)$	Observability Map

Performance Metrics

$\zeta(\tau)$ Efficiency ratio at time τ (combined performance per unit energy)

$E(\tau)$ Total energy consumption at time τ

$S(\tau)$ Total scan certainty at time τ (sum of $c(x, y, \tau)$)

Simulation Parameters

τ Discrete time step in the simulation

τ^{max} Total number of time steps for which the simulation is run

N Number of SAR agents (UAVs/UGVs) in mission

Control and Planning Variables

$\sigma(x, y, \tau)$ Uncertainty reduction rate for cell at position (x, y) at time step τ

$\sigma_{i,j}(x, y, \tau)$ Contribution factor of agent $a_{i,j}$ to total uncertainty reduction rate for cell at (x, y) at time step τ

$r_{i,j}(x, y, z, \tau)$ Euclidean distance between (x_i, y_i) and a cell (x, y) at time step τ

$h_{i,j}(x_i, y_i, \tau)$ Current vertical position of the agent ($h_{i,0} = 0$ →UGVs and $h_{i,1}$ →UAVs)

$u(x, y, \tau)$ Degree of uncertainty of a cell

$c(x, y, \tau)$ Degree of certainty of a cell

$P_V(x, y)$ Probability that cell (x, y) is accurately identified as containing a victim by an agent)

$\rho(x, y, \tau)$ Search priority score for cell (x, y) at time τ

$h^{\text{pred}}_{i,1}(x, y)$ Predicted flying altitude while traversing through a cell at (x, y)

$h^{\text{pred}}_{i,1}(x, y)$ Predicted flying altitude of UAV while traversing through a cell at (x, y)

$L^{\text{p}}_{i,j}$ Length of a path for agent $a_{i,j}$

$v_{i,1}(x, y)$ Constant flying speed of UAV $a_{i,1}$

$v^{\text{pred}}_{i,0}(x, y)$ Predicted ground speed of UGV while traversing through a cell at (x, y)

$\rho(x, y, \tau)$ Search priority score for cell (x, y) at time τ

$\mathbb{P}^{K}_{i,j}$ Set of K shortest paths from the agent's current position for each cell in it's perception field

γ Discount factor for future rewards (in path scoring)

$R^{\text{p}}_{i,j}$ Exploration reward of a path for agent $a_{i,j}$

$T^{\text{p}}_{i,j}$ Travel time of a path for agent $a_{i,j}$

$E^{\text{p}}_{i,j}$ Energy cost of a path for agent $a_{i,j}$

$E^{\text{base}}_{i,j}$ Baseline power requirement for agent $a_{i,j}$

$E^{\text{add}}_{i,j}$ Additional, variable power requirement for $a_{i,j}$

$G^{\text{p}}_{i,j}$ Path grade of a path for agent $a_{i,j}$

Γ, Δ, Θ Weight coefficients for time, exploration, and energy in path grade

$H^{\text{p}}_{i,j}$ Prediction horizon of agent $a_{i,j}$

w_1, w_2, w_3 Global weighting factors in MPC objective (victim detection, coverage, energy)

$A_{i,j}$ Constant representing the agent type in the MPC optimization problem (for UAVs $A_{i,j} = 1$ and $A_{i,j} = 0$ for UGVs

$\mathcal{P}$ Set of paths for N agents; decision variable for the optimisation problem

I_1 Observability threshold for UAVs

I_2 Terrain difficulty threshold for UGVs

List of Figures

1
Executive Summary

This thesis is aimed at improving the efficiency of search-and-rescue (SAR) missions in urban indoor disaster scenarios, where rapid victim detection and comprehensive environmental assessment are vital for saving lives. In order to tackle, the inherent complexity and uncertainty of these disaster environments, an adaptive hierarchical control architecture was designed explicitly for collaborative, multi-agent systems composed of unmanned aerial vehicles and unmanned ground vehicles. The motivation behind this is that the UAV's broader perception and rapid maneuverability enable it to swiftly map expansive areas, while the UGV can focus on detailed searches, navigating through debris-laden or hazardous zones inaccessible to aerial platforms.

The proposed mission-planning framework effectively leverages the distinct and complementary capabilities of UAVs and UGVs to optimize the efficiency of victim detection and enhance situational awareness. This was done by integrating decentralized heuristic controllers employing Fuzzy Logic Control (FLC) with a centralized supervisory controller based on Model Predictive Control (MPC). At the local level, the UAV and UGV independently manage their sensor data, assess environmental conditions, prioritize search areas, and execute paths guided by FLC algorithms. At the higher supervisory level, the centralized MPC controller continuously evaluates real-time environmental data collected by the agents, intervening only when necessary to strategically allocate tasks among agents. This supervisory intervention is driven by real-time assessments of environmental limitations, and it tries to ensure that each agent operates within the scenarios most suited to their capabilities, thus maximizing overall mission optimality.

Extensive simulation experiments conducted within randomized indoor SAR environments demonstrated that the proposed hierarchical cooperative approach consistently outperforms baseline methods, achieving superior victim detection, area coverage, energy efficiency and better overall efficiency outcomes.

Despite these encouraging results, several avenues for future development remain, especially in modeling environmental uncertainties, adaptive and robust control frameworks, which can be scaled to coordinate larger and more diverse multi-robot systems, and perception techniques.

Part I

Scientific Article

Hierarchical MPC-FLC Control Architecture for Energy-Efficient Area Coverage and Victim Detection in Indoor Search-and-Rescue Scenarios via UAV-UGV Collaboration

Shubham Singh

Faculty of Aerospace Engineering, Delft University of Technology, Kluyverweg 1, Delft, 2629 HS, the Netherlands.

Abstract

In time-critical disaster scenarios, heterogeneous robotic teams combining unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) have the potential to enhance situational awareness and improve navigation capabilities in complex indoor environments. This research presents a hierarchical mission planning approach designed for multi-agent search-and-rescue systems (SaRS), explicitly leveraging the distinct sensory and physical capabilities of UAVs and UGVs to maximise victim detection, area coverage, and energy efficiency. At the local control level, each SaR agent utilizes a heuristic controller based on fuzzy logic control (FLC) and the A* pathfinding algorithm, enabling efficient determination of individual targets and paths. At a higher level, a centralised supervisory controller employing model predictive control (MPC) coordinates global mission activities, intervening specifically when an agent's capabilities become suboptimal within certain regions of the environment. Under randomised indoor SaR simulations featuring varied victim distributions, obstacle placements, and environmental conditions—including terrain complexity and visibility, the proposed cooperative approach consistently outperformed baseline methods. Although an exploration-focused heuristic approach (ACS) achieved slightly higher area coverage, the cooperative framework provided comparable coverage while achieving significantly higher victim detection performance, superior consistency, and better overall efficiency. Specialised scenario evaluations further illustrated that the cooperative approach can allocate and execute SaR-related search tasks more effectively, achieving superior operational performance compared to purely local-control-based (selfish) methods. These findings demonstrate that the hierarchical integration of heuristic local control and centralised MPC-based supervision, combined with the complementary capabilities of heterogeneous UAV-UGV teams, results in robust, scalable, and efficient mission planning solutions suitable for complex, uncertain indoor SaR scenarios.

Keywords: *Search and Rescue, Multi-Agent Systems, UAV-UGV Collaboration, Hierarchical Control, Cooperative Search, Mission Planning, Model Predictive Control, Fuzzy Logic Control*

1. Introduction

Over the past few decades, the field of search and rescue (SaR) has witnessed a significant integration of robotic systems, driven by their potential to enhance safety, efficiency, and effectiveness in disaster response scenarios. Deploying autonomous robots in hazardous environments minimises the risks faced by human rescuers, as they can operate in conditions that may be too dangerous or inaccessible for humans [1] [2]. Additionally, automating search tasks allows limited human resources to be allocated to critical areas such as medical aid and logistics [3]. The rapid deployment capabilities of autonomous robots like unmanned aerial vehicles (UAVs) and robust, detailed search capabilities of Unmanned Ground Vehicles (UGVs), enable swift mapping of disaster-struck regions and timely victim detection—a crucial factor in improving survival rates during SaR missions [4]. In such scenarios, situational awareness, which involves gathering vital information about unknown disaster environments like victim locations, hazardous materials, and structural instabilities, is extremely important [5][6]. An important consideration in SaR missions is their varied operational environments. Outdoor SaR typically involves vast, unstructured terrains, such as forests, mountains, or flood zones, where search areas are large. Indoor SaR, on the other hand, takes place in confined, structurally compromised settings like collapsed buildings or tunnels. These environments are often GPS-denied, cluttered with debris, and feature complex, multi-level layouts that challenge navigation, localization, and communication [7] [8] [9].

In this context, multi-robot systems enhance the coverage, speed, and robustness of SaR missions, enabling more thorough and efficient search efforts [10]. They also provide redundancy and perform tasks in parallel, making them well-suited for the demanding conditions of time-sensitive SaR scenarios. Here, a critical distinction exists between homogeneous and heterogeneous multi-robot systems. Homogeneous systems consist of robots with identical capabilities, simplifying coordination. In such systems, multiple identical robots can be assigned to perform the same type of task concurrently, leading to faster completion of the overall mission [11]. In contrast, heterogeneous systems comprise robots with diverse functionalities, such as the combination of UAVs and UGVs. By strategically combining robots with different strengths, a heterogeneous team can address a wider range of challenges encountered in complex disaster scenarios [12]. Evidence of this can be seen in research by Asadi et al. and Sask et al. [13][14], where they design and implement a UAV-UGV collaborative system for monitoring and surveillance of indoor spaces. Earlier, such systems were operated under close human supervision, limiting their autonomy and placing considerable cognitive demands on operators [15]. Challenges such as localisation difficulties and data integration issues often lead to human-induced errors during missions. This has motivated research in autonomous multi-robot SaRS [16], [17], [18], and [19].

Cooperative, multi-agent search approaches are required for efficient SaR and can be categorised as destination-oriented and coverage-oriented search approaches. In scenarios where some prior knowledge of the environment exists, destination-oriented approaches are applied. Their primary goal is to reach pre-determined targets, and they are less focused on gaining additional knowledge of the environment. However, in scenarios where no prior knowledge of the environment exists, as is often the case in disaster situations, gaining new information is crucial. This requires a coverage-oriented search approach. In recent times, significant research has gone into cooperative path planning techniques for multi-robot systems. The objective is to compute a set of feasible paths for each robot in the team, which they must execute to completely scan, explore or survey the structure or environment of interest [20]. Existing literature has explored heuristic approaches like A*, Fuzzy Logic and bio-inspired algorithms like Ant Colony System [21] [22] [23]. Other studies have investigated the application of learning-based approaches like reinforcement learning and neural networks [24] [25] [26]. While these methods can adapt to the environments they were trained in, their performance may degrade when encountering novel or unforeseen situations [27]. However, there is little research on approaches that combine both target/victim-oriented and coverage/area exploration search behaviours. An implementation of a combined approach can be seen in [28], where agents adopt both victim detection and exploratory behaviours. Each agent is assigned a personality type, which determines whether they execute victim detection, area exploration or communication facilitation tasks. Thus, overall, the team can execute multiple objectives. However, here, individual SaR agents are restricted to a set of fixed behaviour types, limiting the flexibility of the search strategy. An improved approach is seen in [29], where the agents dynamically adopt exploration or target visitation roles depending on the environment state and mission requirements.

In the context of control architecture, multi-agent SaR systems employ either centralised or decentralised approaches. Centralised control involves a single entity processing information and directing all agents [30] [31]. Such approaches provide better task-assignment, coordination and global mission performance. Computationally demanding algorithms can be run on a centralised ground station, and tasks/paths can be relayed to dispersed agents. However, centralised control relies heavily on communication between the ground station and agents. This makes them vulnerable to single-point failures [32]. In a decentralised framework, agents in the systems independently make decisions with respect to each other, without the presence of an overarching single point of control [33] [34] [35]. Each robot has its own controller, making it robust to communication failures. However, a decentralised approach, where each agent acts locally, may be sub-optimal for the global mission [36].

Hybrid or hierarchical control architectures aim to combine the strengths of both approaches, coordinating decentralised agents through a centralised framework to balance robustness and efficiency [37]. Such approaches can be particularly useful in multi-robot SaRS, by addressing the challenge of coordinating multiple, decentralised and non-optimal controllers held by individual agents, and adding robustness against communication failures. These advantages have been demonstrated in research by de Koning et al. [38], where they proposed a hierarchical framework using FLC controllers at the local level for real-time decision-making and path prioritisation. FLC controllers use heuristic IF-THEN rules to assign urgency scores to areas within the robot's perception field, which are then evaluated using A* for shortest paths. These candidate paths are graded and selected based on their strategic value for the search mission. At the global level, MPC ensures coordination between robots, avoiding redundant effort by assigning unique search regions to each agent. This combines the real-time adaptability of a local controller with a consideration for global mission optimality. Further, evidence of a hierarchical controller can be found in [39], where a 3-layer hierarchical controller for a multi-robot system was implemented. However, the body of research on hierarchical or hybrid control of heterogeneous robotic teams in SaR applications is still limited and is emerging.

This research focuses on developing a hierarchical mission planning approach tailored for a heterogeneous fleet of UAVs and UGVs in cooperative search operations. By exploiting the complementary sensory and operational capabilities of each agent, the objective is to enhance search performance in a complex, indoor disaster environment. The primary aim is to maximise situational awareness by optimising area coverage and victim detection time while minimising the energy cost. This entails formulating a novel control strategy that dynamically allocates tasks and coordinates movements based on real-time environmental data and agent-specific capabilities. Finally, the report is structured in the following manner: Section 2 lists the primary contributions of the research. Section 3 defines the search problem and related elements. Further, Sections 4 and 5 describe the proposed control framework and case studies to evaluate its performance, respectively. Section 6 discusses the performance of the proposed approach in the simulated cases and compares it with other baseline approaches. Finally, sections 7 and 8 provide a summary of the contents of this paper and some recommendations for future research.

2. Main Contributions

The primary contributions of this research are:

1. A collaborative search strategy is proposed for Indoor SAR missions, utilising a heterogeneous robotic team consisting of UAVs and UGVs. UAVs provide rapid and extensive aerial searches, whereas UGVs can perform detailed ground-level inspections. Despite their individual strengths, each agent type experiences limitations in mobility and perceptual performance under specific environmental conditions, such as reduced visibility or challenging terrain. This research demonstrates that the strategic collaboration between UAVs and UGVs, leveraging their complementary capabilities, can enhance overall search effectiveness and provide system-level robustness to the environmental constraints encountered during SAR missions.
2. A hierarchical control framework, adapted from de Koning's work in [38], is proposed to systematically realise the collaborative UAV-UGV search strategy. The framework combines low-level, decentralised FLC controllers with a high-level, centralised MPC controller. The low-level FLC controller allows rapid, computationally efficient decision-making inspired by human reasoning, and driven by the objectives of area coverage, victim detection, and energy efficiency. At the global supervisory level, the MPC controller maintains situational awareness of environmental conditions as well as the operational capabilities of individual agents. It intervenes by reallocating agents whenever their performance degrades due to environmental conditions, assigning them to areas more compatible with their sensory and locomotion capabilities. Thus, the system benefits from the computational efficiency, robustness, and intuitive decision-making provided by a decentralised FLC layer while simultaneously leveraging the predictive capability, global optimality, and systematic constraint handling inherent to MPC.
3. The proposed hierarchical control framework for energy-efficient area coverage and victim detection by a UAV-UGV team is implemented and validated through extensive simulations in a dynamic indoor environment. Further, a comparison with state-of-the-art methods is presented using multiple benchmarking criteria, including the overall certainty of the environmental mapping, the number of victims successfully detected, the time required for victim detection, and the total energy consumption throughout the mission.

3. Problem Formulation

In this section, the definition and formulation of the mission planning problem for cooperative heterogeneous SaR agents is presented. The mathematical formulations are based on the relations proposed in [38], but have been extended to take into account environmental conditions and the UAV-UGV collaboration.

3.1. SaR Environment

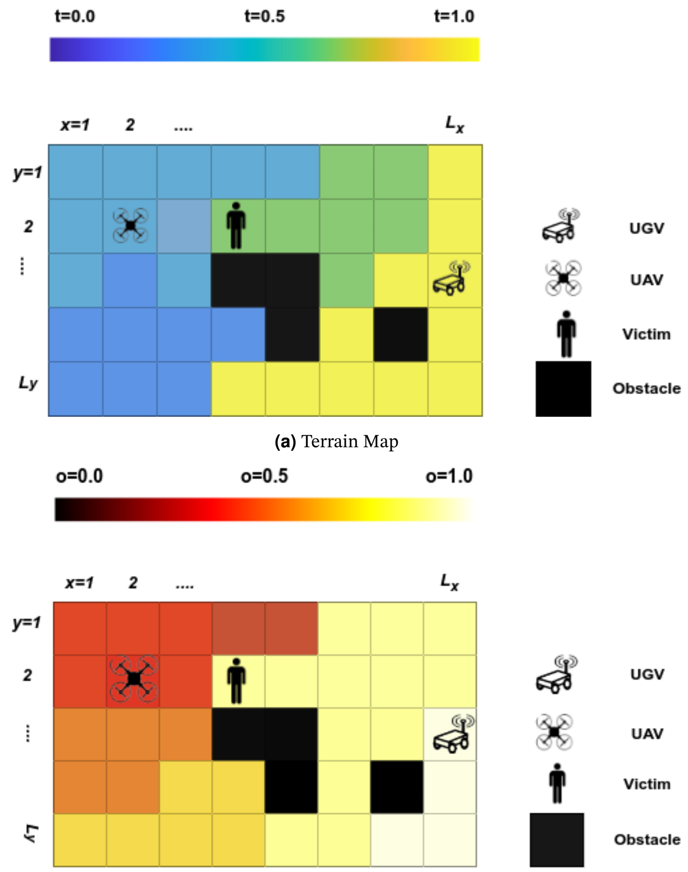
The search-and-rescue (SaR) environment, denoted as E is modeled as a 2D bounded rectangular area consisting of $L_x \times L_y$ cells. Each cell is uniquely identified by its central coordinates (x, y), and is characterised by multiple quantified properties that influence agent navigation, perception, and exploration. These attributes include scan certainty, terrain difficulty, observability, and occupancy status, all of which collectively determine how SaR agents interact with the environment. The scan certainty value, $c(x, y, \tau) \in [0,1]$, represents the degree of information available about a given cell at time step τ. This value is computed based on whether the cell has been previously visited and scanned by an agent, and, if so, the degree to which the acquired sensor data accurately represents the cell's true state. A cell with $c(x, y, \tau) = 0$ is entirely unexplored, while $c(x, y, \tau) = 1$ signifies complete knowledge of its contents. Cells with intermediate values represent regions where uncertainty remains due to limited or imprecise sensing. To track and manage scan certainty across the environment, a scan certainty map, denoted as $\mathcal{C}(\tau)$, is maintained.

Each cell within the environment may either be completely unoccupied or contain static obstacles such as walls, pillars, rubble, etc, as well as dynamic entities, including victims or SaR agents. Some obstacles may render a cell completely inaccessible, while others may only partially obstruct access, allowing agents to navigate through or around them with difficulty. The terrain difficulty index, denoted as $t(x, y) \in [0, 1]$, quantifies the difficulty of traversing a cell at ground level. A value of $t(x, y) = 0$ indicates that the cell is open and fully accessible to ground-based agents (UGVs), whereas $t(x, y) = 1$ corresponds to a heavily congested terrain, such as collapsed structures or dense debris, making the cell entirely impassable. Intermediate values indicate varying degrees of traversal difficulty, which may influence UGVs to modify their paths or expend additional energy to navigate through partially obstructed areas. The terrain index of all cells is recorded in the terrain map, denoted as $\mathcal{T}(\tau)$, which allows SaR agents to assess the feasibility of different paths before execution. This information is particularly crucial for ground-based agents, as terrain difficulty directly impacts their mobility in navigating an environment.

In addition to terrain difficulty, each cell is characterised by an observability index, denoted as $o(x, y) \in [0, 1]$, which represents the ease with which the cell can be perceived using onboard sensors. The observability of a cell is influenced by occlusive elements in the environment, such as debris, smoke, or structural obstacles, which may obscure sensor readings and reduce the accuracy of gathered information. Observability is particularly relevant for UAV operations, as aerial perception is more susceptible to occlusion effects due to the vertical separation between UAVs and ground-level obstacles [40]. Moreover, safety constraints may prevent UAVs from flying at lower altitudes to improve visibility. A value of $o(x, y) = 0$ implies extremely low observability, meaning that aerial scanning is ineffective and information about the cell is difficult to obtain. Conversely, a value of $o(x, y) = 1$ signifies high observability, allowing UAVs to acquire complete and reliable information through rapid scanning at a higher flight altitude. Intermediate values indicate varying degrees of perceptual ease, where a cell may be partially observable. In such cases, the UAV might be required to modify its path or adjust its scanning strategy, i.e, reduce its altitude to reduce the impact of occlusive elements and improve perception. The observability index of all cells is recorded in the observability map, denoted as $\mathcal{O}(\tau)$. This enables agents, more specifically UAVs, to assess the feasibility of different paths based on the mission requirements and mobility constraints.

The occupancy map, denoted as $\mathcal{W}(\tau)$, stores information about cells that are completely inaccessible to the agents and victims due to large static obstacles. With regard to victim occupancy, it is assumed that a cell can be occupied by only a single victim at a time. Depending on the control strategy governing the SaR agents (discussed in in Section 4), these maps can be global, where all agents share information to build a common representation, or local, where each agent relies solely on its own observations. Figures 1a and (1b) show a schematic representation of the environment mapped with the ground truth terrain and the observability conditions of the environment, respectively.

It is important to note that in this research, the environment is dynamic in the sense that the victim positions may change with time step τ (discussed in Section 3.4). However, for simplicity, the physical environmental conditions (like observability, $o(x, y)$, and terrain difficulty, $t(x, y)$) remain static.



(a) Terrain Map

(b) Observability Map

Figure 1. Simulated SAR environment with: (a) Terrain Map, (b) Observability Map

3.2. SaR Agents

This research utilizes a multi-agent system composed of N heterogeneous agents, including rotary-wing UAVs and ground-based UGVs. Each agent is denoted as $a_{i,j}$, where $i \in [1, N]$ represents the agent index, and j indicates the agent type ($j = 1$ for UAVs and $j = 0$ for UGVs). The agents are characterised by varying sensing capabilities.Firstly, SaR agents may have different perception radii $r^{\mathrm{p}}_{i,j}$, which defines the spatial extent to which an agent can scan and acquire information. The perception field of agent i, denoted as $E_i(x_i, y_i, r^{\mathrm{p}}_{i,j})$, consists of all cells within a circular region of radius $r^{\mathrm{p}}_{i,j}$, centered at the agent's current position, (x_i, y_i). The perception field of each agent is a subset of the SaR environment, i.e., $E_i(x_i, y_i, r^{\mathrm{p}}_{i,j}) \in E$. Secondly, the *perceptual uncertainty reduction rate* $\eta_{i,j}$ for various SaR agents may be different, which is a parameter that quantifies the accuracy of an agent's sensor [38]. Lastly, environmental conditions introduce occlusion effects, which impact the perception capabilities of SaR agents to different extents. To model this variation, an *occlusion sensitivity factor* $\alpha_{i,j}$ is introduced. This parameter quantifies how strongly an agent's sensing ability is degraded by occlusions such as debris, smoke, or structural obstacles. It can have a value belonging to the set $(0, 1]$. Agents with higher $\alpha_{i,j}$ values (specifically UAVs) experience significant perception loss in occluded regions, while those with lower $\alpha_{i,j}$ values (such as UGVs) perceive better due to their proximity to the ground.

The SaR agents differ in their movement capabilities, which are influenced by their mode of locomotion and the environmental conditions they encounter. Each agent's position at any given time step is represented as a discrete cell within the grid-based environment, E. For UGVs, movement is heavily influenced by terrain, as rough or obstructed ground surfaces affect their navigability. It is assumed that a UGV can advance by at most one cell per time step. In contrast, UAVs operate independently of ground conditions and are assumed to be capable of moving at their maximum forward velocity. As a result, UAVs can traverse up to two cells per time step. All agents can move in any of the 8 directions (north, northeast, east, southeast, south, southwest, west, and northwest), as illustrated in Figure 2.

It is assumed that the agents can freely move about their central axis, inside the current cell. Hence, an agent can move in any of the 8 possible directions, unconstrained by its orientation.

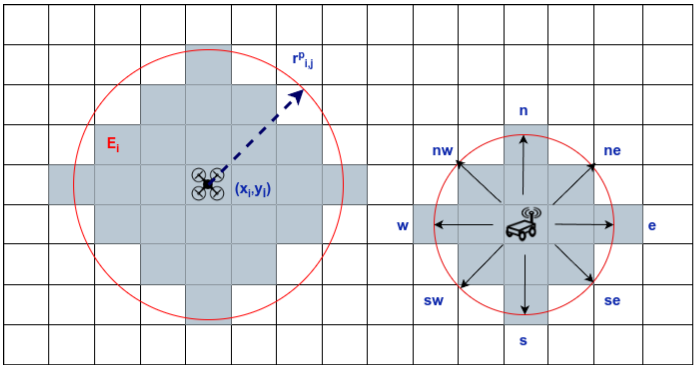


Figure 2. Perception field and possible directions for movement of an agent

3.3. Sensory Capabilities of SaR Agents

Each SaR agent is equipped with onboard optical cameras used for victim detection and acquisition of visual data from the environment. The gathered information is subsequently recorded in the corresponding maps for scan certainty ($\mathcal{C}$), observability ($\mathcal{O}$), and terrain difficulty ($\mathcal{T}$). These maps can be either global or local, depending on the adopted search strategy.

SaR agents may operate with inherent sensor imperfections, meaning that a single scan does not guarantee complete knowledge of all cells within an agent's perception field. In this research, we have considered three primary sources of sensory imperfections: (I) a single scan at a given time step may fail to provide complete certainty about a cell due to the intrinsic limitations of the agent's sensors. To account for this, Dempster's rule of combination [41] is applied, allowing information from multiple scans over time to gradually reduce *uncertainty*. This reduction is characterized by the perceptual uncertainty reduction rate $\eta_{i,j} \in (0, 1]$ associated with agent $a_{i,j}$. (II) Although all cells within the perception field $E_i(x_i, y_i, r^{\mathrm{p}}_{i,j})$ of agent $a_{i,j}$ are scanned by its (imperfect) sensor, the efficacy of the scan is modulated by the distance between the agent and the target cell. Cells located further from the agent experience a lower certainty update rate due to decreasing sensor precision with increasing range. (III) The physical conditions of a cell, such as the presence of rubble, smoke, or structural obstructions, directly affect an agent's ability to accurately sense and map its environment. A higher density of such occlusive elements translates into lower observability and, subsequently, a lower rate of change of scan certainty. The degree to which the perception capability of agent $a_{i,j}$ is degraded by environmental occlusion, or in other words, is impacted by observability conditions, is represented by the *occlusion sensitivity factor*, $\alpha_{i,j}$. The following equation, adapted from [38], mathematically expresses the *uncertainty reduction rate* $\sigma(x, y, \tau)$ for a cell (x, y):

$$\sigma(x, y, \tau) = \prod_{i=1, j=0|1}^{N} \max\left\{\sigma_{i,j}(x, y, \tau), 1\right\} \tag{1}$$

with:

$$\sigma_{i,j}(x, y, \tau) = 1 - (1 - \eta_{i,j}) \cdot e^{-\left(r_{i,j}(x,y,z,\tau) + {\alpha_{i,j}}^{h_{i,j}(x_i, y_i, \tau) \cdot o(x,y)}\right)} \cdot \frac{1 - sign\left(r_{i,j}(x, y, z, \tau) - r^{\mathrm{p}}_{i,j}\right)}{2} \tag{2}$$

where the exponential decay term accounts for the effect of both agent-cell proximity and observability constraints on scan effectiveness. The Euclidean distance $r_{i,j}(x, y, z, \tau)$ models the proximity effect, where scan certainty diminishes as the target cell is located farther from the agent. The effect of observability constraints is represented by the second term, where the observability index $o(x, y)$ is modulated by the occlusion sensitivity factor $\alpha_{i,j}$.

As discussed earlier, UAV search agents will have a higher value for $\alpha_{i,j}$, indicating that their ability to scan has greater sensitivity to the observability conditions. The factor $h_{i,j}(x_i, y_i, \tau)$ represents the current vertical position of the agent (here $j = 0$ for UGV and $j = 1$ for UAV) and its product with the $o(x, y)$ is meant to indicate an altitude-observability interaction. As the search agent (specifically UAVs) increases its operational altitude [42], its ability to accurately scan a given cell may diminish. This may be due to a reduction in sensor resolution, making it difficult to identify partially occluded objects (such as victims in regions of extremely high smoke density). In regions that have high observability, the UAV may choose to fly at a higher altitude to enhance operational safety and energy efficiency (discussed in Section 4), as the reduction in scan accuracy due to the proximal and environmental occlusion effects (quantified by observability index) is relatively minor. Alternatively, in low-observability regions where occlusion is high, the UAV might choose to reduce its flight altitude as the loss in scan accuracy may be unreasonably high at higher flight altitudes. In the case of UGVs, this factor is significantly less relevant since its vertical separation from the ground level is minimal (i.e., low $h_{i,0}(x_i, y_i, \tau)$), and it can overcome observability constraints to a certain extent by performing much closer inspections of regions of interest. Mathematically, this translates to a low value for $\alpha_{i,0}$.

Note that $r_{i,j}(x, y, z, \tau) = r_{i,0}(x, y, 0, \tau)$ in the case of UGVs, as their vertical displacement is negligible. In (2), $\sigma_{i,j}(x, y, \tau)$ denotes the individual contribution of SaR agent $a_{i,j}$ to the uncertainty reduction rate for cell (x, y). According to the proposed model, when $r_{i,j}(x, y, z, \tau) = 0$, i.e., the agent is located at the cell, the corresponding uncertainty reduction rate is determined by both the observability constraints and the perceptual reduction rate $\eta_{i,j}$. If $r_{i,j}(x, y, z, \tau) \geq r^{\mathrm{p}}_{i,j}$, meaning the cell lies outside the agent's perception range, the contribution is zero. Equation (1) captures the aggregate effect on scan certainty, of all N SaR agents operating within the environment. To ensure that agents located too far from a given cell do not affect the overall uncertainty update, the `max` function is employed in (1).

Finally, the updated scan certainty for cell (x, y) is computed as:

$$c(x, y, \tau + 1) = 1 - u(x, y, \tau + 1) \tag{3}$$

with:

$$u(x,y,\tau+1) = \sigma(x,y,\tau)\cdot u(x,y,\tau) \;\; \forall\,(x,y)\in E_i(x_i,y_i,r^{\mathrm{p}}_{i,j}) \qquad (4)$$

Here, the degree of uncertainty of a cell $u(x,y,\tau)$ forms the complement of $c(x,y,\tau)$.

3.4. Victim Modelling

The number, and location of the victims are initially unknown to the SaR robots. The victims follow a random pattern of movement, i.e., victim v with position $(x^{\mathrm{V}}_v(\tau), y^{\mathrm{V}}_v(\tau))$ at time step τ may remain in the current cell with probability p_s or move to a neighbouring cell with probability $1-p_s$. This results in an equal probability:

$$p^{\mathrm{m}}_v = \frac{(1-p_s)}{n^{\mathrm{free}}_v(x^{\mathrm{V}}_v(\tau), y^{\mathrm{V}}_v(\tau), \tau)} \qquad (5)$$

where $n^{\mathrm{free}}_v(x^{\mathrm{V}}_v(\tau), y^{\mathrm{V}}_v(\tau), \tau)$ is the number of free neighbouring cells for cell $(x^{\mathrm{V}}_v(\tau), y^{\mathrm{V}}_v(\tau))$ at time step τ. Thus, the relation in (5) indicates the probability of moving to any one of the free neighbouring cells adjacent to the victim's current position.
In this study, a victim is considered detected by a SaR agent only when both the agent and the victim occupy the same cell. Upon detection, the agent records the victim's location and the corresponding time step in its local victim map, denoted as $\mathcal{V}_i(\tau)$. Since this information is stored locally, each SaR agent retains knowledge exclusively of the victims it has individually identified.

4. Hierarchical Control System

This section outlines the hierarchical control architecture that determines how the SaR agents carry out their search behaviour, and the control strategies employed at different levels. The proposed controller operates at two levels, as illustrated in Figure 3, ensuring that agents perform search operations efficiently while learning about the environment.

At the lower level, a decentralised control system manages the local search behaviour of each SaR agent autonomously (see Section 4.1 for details). These independent controllers dictate the movement and scanning strategies of their respective agents based on local sensory data, within the agent's immediate surroundings. At the higher level, a centralised supervisory controller oversees the global coordination of the entire fleet of SaR agents. The primary role of this supervisory controller is to strategically allocate agents to regions where their mode of translation and sensory capabilities are most effective. Specifically, UAVs are directed toward high-observability regions, where aerial scanning is most efficient, while UGVs are assigned to terrain-accessible areas, where ground traversal is feasible (discussed in Section 3.1). This strategic coordination ensures that agents operate in optimal regions, enhancing the overall efficiency of the search-and-rescue mission. A key characteristic of this control architecture is that SaR agents communicate exclusively with the supervisory controller rather than directly with each other. This design integrates the advantages of both centralised and decentralised control approaches. While the local controllers enable SaR agents to autonomously navigate and adapt to dynamic conditions, the supervisory controller ensures global mission coordination, optimising resource allocation and constraint satisfaction. This hybrid control structure enhances system robustness, as local agents can continue operating even in the event of communication failures, while computationally intensive tasks, such as mission-wide decision-making, are handled at the centralised control level.

4.1. Local Controller

The local controller of each SaR agent functions in two primary stages. In the first stage, the agent utilises its local sensory inputs to generate a priority map over its perception field. This map assesses the significance of visiting each cell and estimates the associated effort or energy cost required for traversal. This priority map provides a basis for decision-making, guiding the agent toward areas where information gain is maximised while considering traversal constraints. Once the priority map is established, the controller selects the path that yields the highest net benefit, quantified through a path-grading mechanism. Each potential path is assigned a grade, reflecting how favourable it is relative to the mission's objectives. These objectives include maximising search efficiency and coverage while minimising both time and energy expenditure. As a result, the path grading process incorporates the following three criteria:

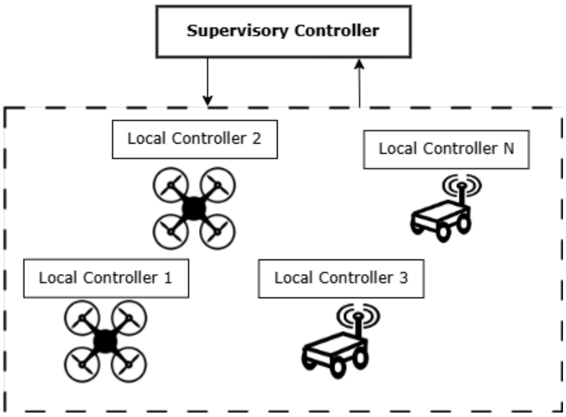


Figure 3. Hierarchical architecture of the cooperative mission planning controller

1. **Reduction of time:** The SaR agents prioritise reaching targets of interest in the shortest possible time, thereby reducing the overall duration of the SaR mission.
2. **Increase of exploration:** The agents aim to scan the maximum number of previously unexplored cells as possible along it's route, increasing the total area covered during the mission.
3. **Reduction of energy consumption:**The agents seek to reduce energy expenditure while navigating by avoiding energy-intensive scenarios such as UGV movement through difficult terrain or UAV operations in low-observability regions.

These three path grading criteria introduce conflicting priorities in the agent's decision-making process. The time efficiency objective encourages a destination-oriented approach, selecting the shortest path to a target cell. Conversely, the exploration objective promotes a coverage-oriented strategy, favouring paths that pass through multiple unexplored regions, even at the cost of increasing travel time. The reduction of energy consumption criterion further complicates this trade-off, as certain paths that optimise time or coverage may require higher energy expenditure, particularly in high-difficulty terrain for UGVs or low-observability regions for UAVs. The local controller is designed to balance these competing objectives, ensuring that the selected path balances speed, coverage, and energy efficiency to enhance overall SaR mission performance.

4.1.1. Search Priority Assignment

The first component of the local controller of each SaR agent is the task priority allocation mechanism. In this research, the primary objective of SaR agents is environmental scanning. Accordingly, task priority is defined as the degree of urgency associated with scanning a specific cell within the environment. A priority score is assigned to each cell within the perception field $E_i(x_i,y_i,r^{\mathrm{p}}_{i,j})$ of an agent using FLC. The choice of FLC as the control approach is motivated by its computational efficiency and its ability to model human decision-making in a mathematically structured manner. Given that SaR agents operate with limited onboard computational resources, the control method must be both lightweight and effective. In real-world rescue operations, human operators are still primarily responsible for resource allocation and mission planning, making it beneficial for an autonomous SaR system to replicate human-like reasoning. For this, the local controller utilises a rule-based Mamdani fuzzy inference system (FIS). It relies on a set of M if-then rules, which define the agent's priority assignment strategy and are structured as follows:

$$\begin{aligned}\mathcal{R}_m\ :\ &\text{If } P_V(x,y) \text{ is } A_{m,1}\ , c(x,y,\tau) \text{ is } A_{m,2} \text{ and } q^{\mathrm{e}}_j(x,y) \text{ is } A_{m,3}\\ &\text{then } \rho(x,y,\tau) \text{ is } B_{m,1},\ \text{and } \chi(x,y) \text{ is } B_{m,2}\\ &m=1,2,\dots,M,\quad (x,y)\notin E_i(x_i,y_i,r^{\mathrm{p}}_{i,j}).\end{aligned} \tag{6}$$

In the proposed fuzzy inference system (FIS), both the antecedent propositions $A_{m,a}$ and the consequent propositions $B_{m,b}$—where $a\in\{1,2,3\}$ and $b\in\{1,2\}$, are represented as fuzzy sets, each mathematically defined by its respective membership function. The task priority allocation module processes three primary inputs, all of which are derived from the local sensory knowledge of each SaR agent. Since agents do not share perception data, each SaR agent operates based solely on information available in its scan certainty map $C_i(\tau)$, victim map $\mathcal{V}_i(\tau)$ and the relevant environmental condition map, $Q^{\mathrm{e}}_j(\tau)$. For UAVs, $Q^{\mathrm{e}}_j(\tau)$ corresponds to the *observability map*, $\mathcal{O}(\tau)$ and $q^{\mathrm{e}}_j(x,y)$ corresponds to $o(x,y)$. While for UGVs it corresponds to the *terrain difficulty map*, $t(x,y)$ and $q^{\mathrm{e}}_j(x,y)$ corresponds to $t(x,y)$. As a result, individual SaR agents lack direct knowledge of victims detected or areas scanned by other agents, ensuring that each agent's priority assignment is based entirely on its localised perception.

The first input is the victim probability P_V, i.e, the probability that a particular cell is accurately identified as containing a victim. In other words, the degree of confidence with which that agent identifies a victim-occupied cell. Cells expected to be unoccupied by victims have a probability of zero, whereas cells where a victim is detected, are assigned a small positive value, which is distance-dependent to account for potential proximity-related sensor inaccuracies, as seen in equation (7).

$$P_V(x,y) = -\left(\frac{r(x_i,y_i,x,y)}{r^{\mathrm{p}}_{i,j}}\right)^2 + 1 \quad \forall (x,y)\in E_i(x_i,y_i,r^{\mathrm{p}}_{i,j}) \tag{7}$$

where $r(x_i,y_i,x,y)$ is the distance from the agent's position (x_i,y_i) to a cell (x,y) within the current perception field of the agent, $E_i(x_i,y_i,r^{\mathrm{p}}_{i,j})$. Thus, according to this formulation, closer cells receive a higher probability value, reflecting the agent's increased confidence in detecting victims at shorter distances [38]. The second input is scan status $c(x,y,\tau)$ and has been discussed in Section 3.3. The third input, the environmental condition parameter $q^{\mathrm{e}}_i(x,y)$, accounts for agent-specific environmental constraints that influence its movement. As mentioned earlier, for UAVs this input is the observability index $o(x,y)$ and for UGVs it is the terrain difficulty index $t(x,y)$.

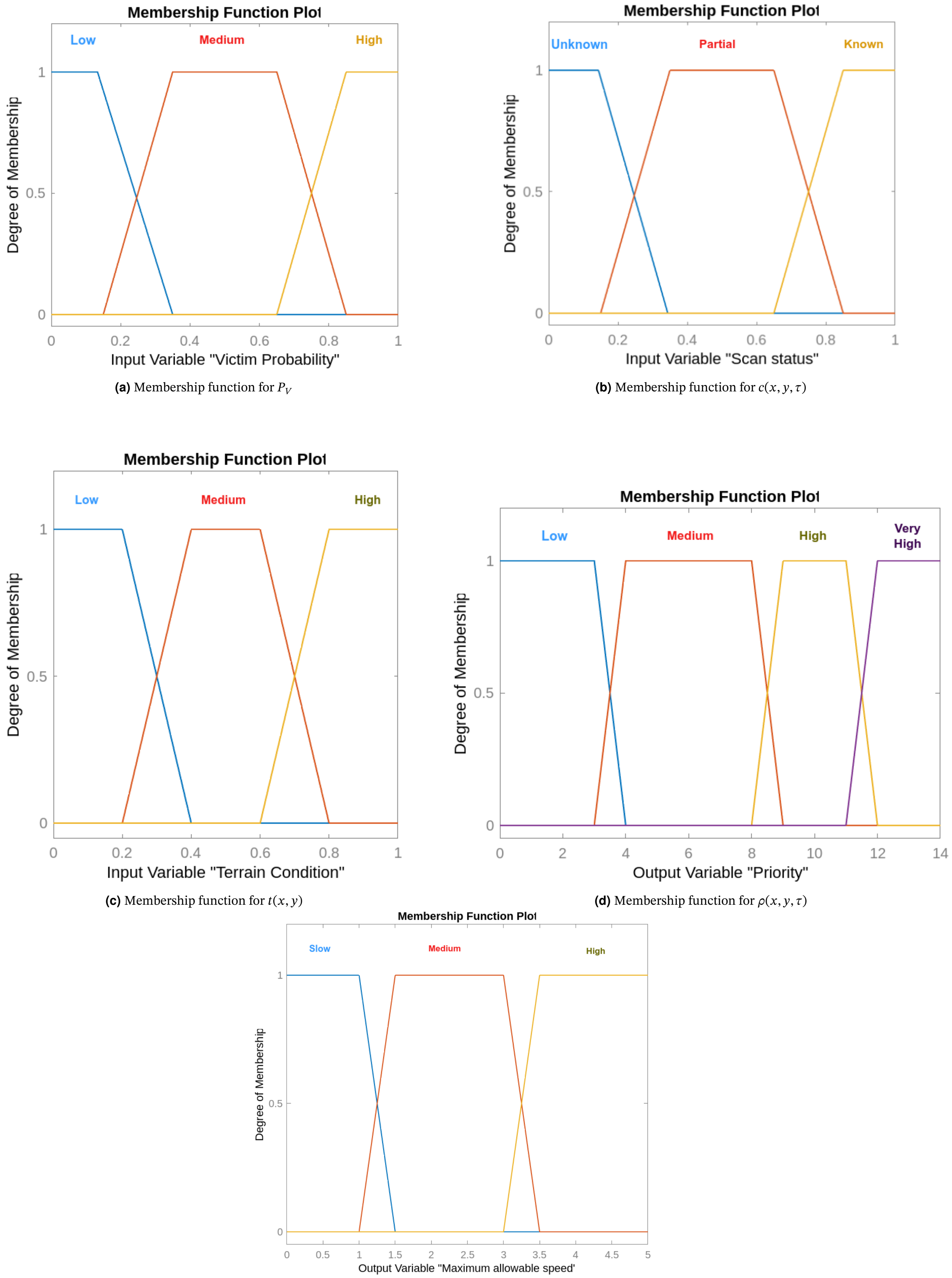
Using these inputs, the FIS generates two outputs. First, we get the search priority score $\rho(x,y,\tau)$ for each cell in $E_i(x_i,y_i,r^{\mathrm{p}}_{i,j})$, which represents the urgency to scan that cell. The search priority is determined solely by victim probability and scan status, ensuring that, in general, areas with a higher likelihood of containing victims and lower scan certainty are prioritised for visitation. The second output is the environmental constraint that the cell, based on the prevailing environmental conditions, imposes on the agent. For UGVs, this is in the form of predicted translation velocity ($v^{\mathrm{pred}}_{i,0}(x,y)$), while traversing through a cell at (x,y). A higher terrain index results in difficulty in moving forward and, thus, a lower ground translation velocity. For UAVs, the constraint is on the predicted flying altitude ($h^{\mathrm{pred}}_{i,1}(x,y)$) while traversing through a cell at (x,y). A higher observability value permits operation at greater altitudes, whereas low observability necessitates flying at lower altitudes to improve scan accuracy. All the inputs and the second output ($h^{\mathrm{pred}}_{i,1}(x,y)$ for UAV and $v^{\mathrm{pred}}_{i,0}(x,y)$ for UGV), have the fuzzy membership functions as *Low, Medium, and High*, while the priority output ($\rho(x,y,\tau)$) has the membership functions as *Very High, High, Medium, Low and Very Low*. These can be seen in Figure 4 for the UGV and Figure 5 for the UAV. The Trapezoidal membership function was selected because it allows an input variable to belong fully to a particular fuzzy category within a range of values rather than at a single precise point. This is beneficial as the variables being handled here (victim probability and environmental conditions) are often uncertain. Additionally, it provides a smooth transition between membership levels. This is important, since, one of the outputs is a control action ($h^{\mathrm{pred}}_{i,1}(x,y)$ for UAVs and $v^{\mathrm{pred}}_{i,0}(x,y)$ for UGVs). Further, based on the rule structure in (6), the complete rule base for the UAV and the UGV can be seen in Table 1 and Table 2, respectively. It is important to note that the search priority score is independent of the environmental conditions; rather, the environmental constraint influences the mission time and energy cost, which are separate grading criteria within the local controller. This will be discussed in Section 4.1.3. Finally, cells in $\mathcal{W}(\tau)$ are assigned a priority score of zero, so that SaR agents do not attempt to navigate impassable regions.

4.1.2. Path Planning

Next, we consider the path planning module, which is the second component of the local mission planning controller. It determines the optimal movement strategy for the SaR agent. Given the time constrained nature of SaR missions, the SaR agent is partially destination-oriented, requiring an approach that prioritises shortest-path navigation. However, since the SaR agent must also maximise exploration, its path planning considers all cells within its perception field as potential target destinations. The first stage of the path planning process identifies the globally shortest path p^1_i from the SaR agent's position, (x_i,y_i), at current time-step τ, represented by $s_i(x_i,y_i,\tau)$, to any given potential target cell $f_i(x,y)$ within $E_i(x_i,y_i,r^{\mathrm{p}}_{i,j})$. To achieve this, the controller employs the A* search approach [43], which is an extension of Dijkstra's algorithm [44]. A* search limits the number of evaluated paths. Instead of exhaustively exploring all possible paths, it incorporates a heuristic function that prioritises paths leading roughly towards the goal while disregarding paths that diverge significantly. This significantly decreases the computational cost and time required to identify the optimal path, particularly in grid-based environments where distances between cells are uniform. Once the shortest path is determined, the next step is to identify alternative paths that are comparable in length but traverse different areas within the perception field. This aligns with the coverage-oriented objective of the SaR agent. To achieve this, the controller uses Yen's algorithm [45], which efficiently computes K shortest paths $\mathbb{P}^K_{i,j}=\{p^1_i,p^2_i,\dots,p^K_i\}$ for agent $a_{i,j}$. This algorithm systematically generates these alternative paths by iteratively modifying the previously found shortest path [38]. It does so by systematically closing subsequent nodes within the previously found shortest path p^{k-1}_i, evaluating the resulting alternative routes, and storing them in a set. The next shortest path p^k_i, is selected from this set, and the process repeats until K distinct shortest paths are identified. These shortest paths omit cells present in $\mathcal{W}(\tau)$, in order to avoid obstacles.

4.1.3. Path Grading

After identifying, K shortest paths for each potential target cell (all cells in $E_i(x_i,y_i,r^{\mathrm{p}}_{i,j})$), the next step is to evaluate and grade each target-path combination to determine the path with maximum reward for the SaR agent. As previously discussed, the path grade reflects the favourability of a given path based on three key factors: travel time, exploration gain, and energy cost. These factors



(a) Membership function for P_V

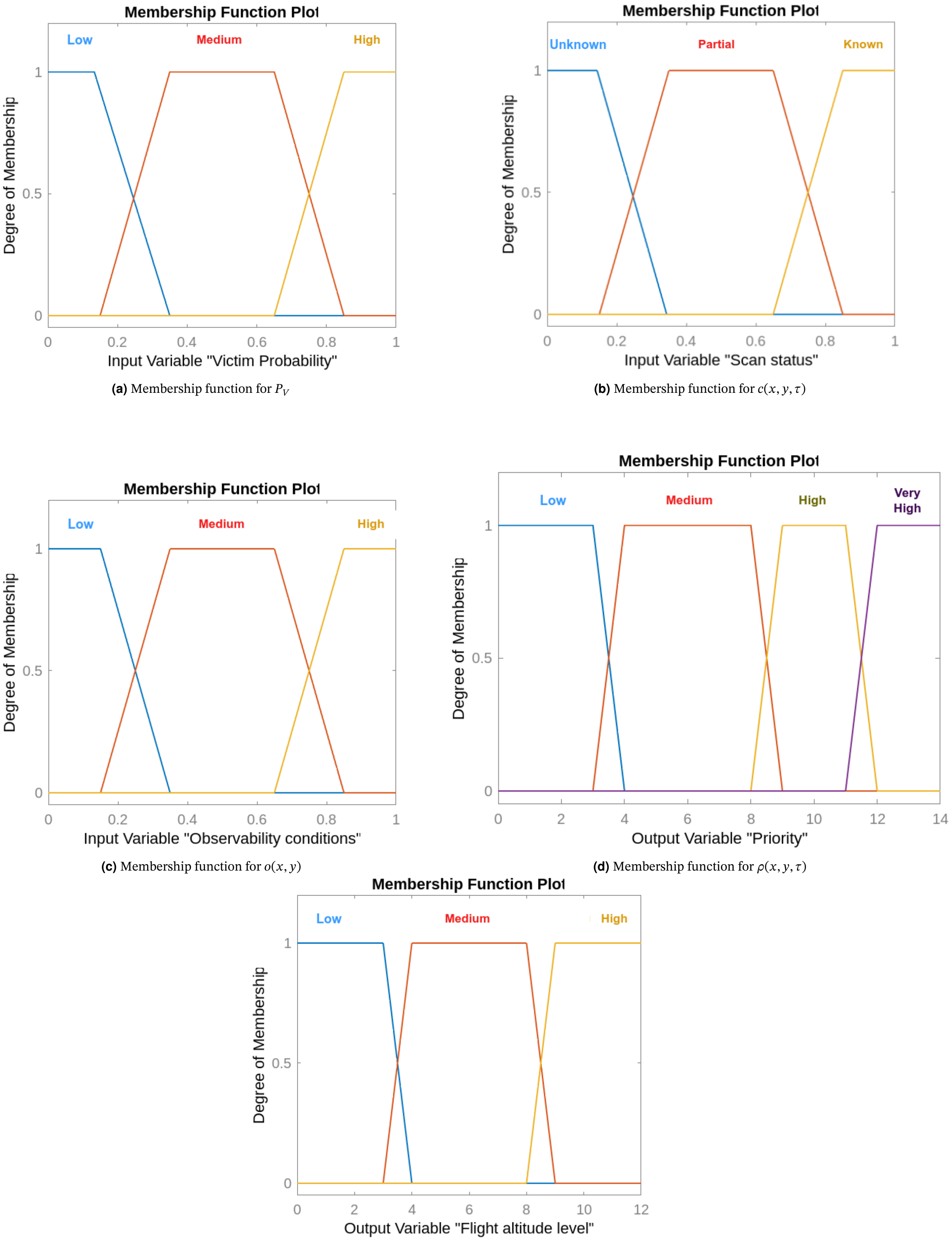
(b) Membership function for $c(x, y, \tau)$

(c) Membership function for $t(x, y)$

(d) Membership function for $\rho(x, y, \tau)$

(e) Membership function for $v_{i,0}^{\text{pred}}$

Figure 4. Input and Output Membership Functions for UGV



(a) Membership function for P_V

(b) Membership function for $c(x, y, \tau)$

(c) Membership function for $o(x, y)$

(d) Membership function for $\rho(x, y, \tau)$

(e) Membership function for $h_{i,1}^{\text{pred}}(x, y)$

Figure 5. Input and Output Membership Functions for UAV

$\mathcal{R}_i$	P_v	$c(x,y,\tau)$	$o(x,y)$	$\rho(x,y,\tau)$	$h^{\text{pred}}_{i,1}(x,y)$
1	High	Unknown	High	Very High	High
2	High	Unknown	Medium	Very High	Medium
3	High	Unknown	Low	Very High	Low
4	High	Partial	High	High	High
5	High	Partial	Medium	High	Medium
6	High	Partial	Low	High	Low
7	High	Known	High	Medium	High
8	High	Known	Medium	Medium	Medium
9	High	Known	Low	Medium	Low
10	Medium	Unknown	High	High	High
11	Medium	Unknown	Medium	High	Medium
12	Medium	Unknown	Low	High	Low
13	Medium	Partial	High	Medium	High
14	Medium	Partial	Medium	Medium	Medium
15	Medium	Partial	Low	Medium	Low
16	Medium	Known	High	Low	High
17	Medium	Known	Medium	Low	Medium
18	Medium	Known	Low	Low	Low
19	Low	Unknown	High	Medium	High
20	Low	Unknown	Medium	Medium	Medium
21	Low	Unknown	Low	Medium	Low
22	Low	Partial	High	Low	High
23	Low	Partial	Medium	Low	Medium
24	Low	Partial	Low	Low	Low
25	Low	Known	High	Very Low	High
26	Low	Known	Medium	Very Low	Medium
27	Low	Known	Low	Very Low	Low

Table 1. FIS rule base for UAV

$\mathcal{R}_i$	P_v	$c(x,y,\tau)$	$t(x,y)$	$\rho(x,y,\tau)$	$v^{\text{pred}}_{i,0}(x,y)$
1	High	Unknown	High	Very High	Slow
2	High	Unknown	Medium	Very High	Medium
3	High	Unknown	Low	Very High	Fast
4	High	Partial	High	High	Slow
5	High	Partial	Medium	High	Medium
6	High	Partial	Low	High	Fast
7	High	Known	High	Medium	Slow
8	High	Known	Medium	Medium	Medium
9	High	Known	Low	Medium	Fast
10	Medium	Unknown	High	High	Slow
11	Medium	Unknown	Medium	High	Medium
12	Medium	Unknown	Low	High	Fast
13	Medium	Partial	High	Medium	Slow
14	Medium	Partial	Medium	Medium	Medium
15	Medium	Partial	Low	Medium	Fast
16	Medium	Known	High	Low	Slow
17	Medium	Known	Medium	Low	Medium
18	Medium	Known	Low	Low	Fast
19	Low	Unknown	High	Medium	Slow
20	Low	Unknown	Medium	Medium	Medium
21	Low	Unknown	Low	Medium	Fast
22	Low	Partial	High	Low	Slow
23	Low	Partial	Medium	Low	Medium
24	Low	Partial	Low	Low	Fast
25	Low	Known	High	Very Low	Slow
26	Low	Known	Medium	Very Low	Medium
27	Low	Known	Low	Very Low	Fast

Table 2. FIS rule base for UGV

introduce conflicting objectives. For instance, while the shortest path minimises travel time, it may lead the agent through low-priority regions, missing opportunities for better exploration. Conversely, a path that maximises exploration may significantly increase mission duration and energy consumption if it passes through a sub-optimal region for the specific agent's operation. These factors are expressed mathematically to logically grade a path.

The *time* required to traverse a path depends on both the path length and the agent's predicted speed of movement along the path. For UAVs, speed $v^{\text{pred}}_{i,1}$ is assumed to be constant at all positions (x,y) and time steps τ, as they can move unhindered at maximum velocity across the environment. Thus, the path duration $T^{\mathrm{p}}_{i,1}$ is simply, $\frac{L^{\mathrm{p}}_{i,1}}{v_{i,1}}$, where $L^{\mathrm{p}}_{i,1}$ is the Euclidean distance of the path, and $v_{i,1}$ is the constant velocity of traversal, for agent $a_{i,1}$. However, movement speed for UGVs is terrain-dependent, as difficult terrain reduces mobility, whereas easily traversable terrain allows for higher speed. Therefore, the predicted time required to traverse a certain path, for an agent $a_{i,0}$, is computed as:

$$T^{\mathrm{p}}_{i,0} = \frac{L^{\mathrm{p}}_{i,0}}{v^{\text{average}}_{i,0}} \tag{8}$$

where $v^{\text{average}}_{i,0}$ for the path is computed as:

$$v^{\text{average}}_{i,0} = \frac{1}{N^{\text{cells}}} \sum_{i=1}^{N^{cells}} v^{\text{pred}}_{i,0}(x,y) \tag{9}$$

As mentioned earlier, $v^{\text{pred}}_{i,0}(x,y)$ is the predicted UGV velocity (obtained from its FIS) for all cells ($i = 1, \ldots, N^{\text{cells}}$) belonging to the path. Secondly, the extent of environmental exploration along a path is considered as a grading criterion. While this can be done by counting the number of cells that the agent visits along the way, this could lead the agent to visit many low-priority cells before finally reaching the goal cell. To mitigate this, the discounted return framework is used to ensure that paths prioritising visiting high-value cells earlier receive higher scores. The discounted return $R^{\mathrm{p}}_{i,j}$ of a path is computed in the following manner:

$$R^{\mathrm{p}}_{i,j} = \sum_{\tau=0}^{H^{\mathrm{p}}_{i,j}} \gamma^{\tau} \rho(x,y,\tau) \tag{10}$$

The discounted return represents the cumulative reward obtained by an agent over a sequence of actions. In this context, the actions executed by agent $a_{i,j}$ correspond to the sequence of cells visited within its prediction horizon $H^{\mathrm{p}}_{i,j}$, with each cell associated with a priority score $\rho(x,y,\tau)$, which serves as the immediate reward. In this formulations, future rewards are diminished using an exponential discount factor γ^{τ}, where $\gamma \in [0,1]$. The idea is that paths that compel the agent to visit more high-priority cells sooner are prioritised.

Thirdly, paths are graded based on the energy the agent has to expend to travel along it, as excessive energy expenditure reduces mission endurance. In this research, a simplified energy model is designed to reflect the unique operational constraints of both UAVs and UGVs. For UAVs, energy consumption is heavily influenced by flight altitude, which is dictated by observability conditions along the path. In low-observability regions, UAVs must fly at lower altitudes to mitigate environmental occlusion effects (e.g., due to smoke, debris) on sensing, thereby improving scan accuracy. This is based on a general understanding that the impact of occlusion effects on the accuracy of sensors like cameras and LIDARs increases with distance. However, flying at lower altitudes incurs higher energy costs due to increased computational demand for onboard processing of dense visual data. This may be due to the increase in the density of features of interest in a narrower field of view. In addition to this, the density of obstacles to be avoided while flying is likely to be higher at a lower altitude, which would translate into more energy being expended for avoidance maneuvers. The following relation is inspired from [46] and shows a simplified, artificial model for the energy cost of the path for a UAV:

$$E_{i,1}^{p} = E_{i,1}^{\text{base}} + \sum_{i=1}^{N^{cells}} \frac{E_{i,1}^{\text{add}}}{h_{i,1}^{\text{pred}}(x,y)} \tag{11}$$

where $E_{i,1}^{\text{base}}$ is the baseline power consumed by the UAV to sustain flight. Moreover, $E_{i,1}^{\text{add}}$ represents additional power demands for scanning, obstacle avoidance, and onboard processing. This is modulated according to the UAV's predicted flight altitude, $h_{i,1}^{\text{pred}}(x,y)$. The summation of this modulated $E_{i,1}^{\text{add}}$ is taken for all cells ($i = 1, \dots, N^{\text{cells}}$) belonging to the path and further summed with the fixed baseline power. Similarly, for the UGV energy consumption is determined by terrain difficulty, as navigating through rough terrain requires additional mechanical effort. The following simplified relation, inspired from formulations in [47] and [48], is used to model the energy cost:

$$E_{i,0}^{p} = E_{i,0}^{\text{base}} + E_{i,0}^{\text{add}} \tag{12}$$

where:

$$E_{i,0}^{\text{add}} = \sum_{i=1}^{N^{cells}} \frac{1}{1 - t(x,y)} \tag{13}$$

As in equation (11), $E_{i,0}^{\text{base}}$ represents the baseline power for the UGV to remain in operation. The second term represents the additional variable energy required to traverse the terrain. For cells with highly difficult terrain, i.e, high $t(x,y)$ value, the additional energy requirement is higher. The overall objective of this factor is to prioritise paths which are energy efficient.

With all inputs defined, the final path grade $G_{i,j}^{\text{p}}$ of each agent is computed as:

$$G_{i,j}^{\text{p}} = -\Gamma \cdot T_{i,j}^{\text{p}} + \Delta \cdot R_{i,j}^{\text{p}} - \Theta \cdot E_{i,j}^{p} \tag{14}$$

Here, Γ, Δ and Θ are weight coefficients that balance the influence of time, exploration, and energy cost, respectively. It should be noted that visiting high-priority cells leads to a higher overall path return, $R_{i,j}^{\text{p}}$ and subsequently, a higher grade is assigned to the path. At the same time, $T_{i,j}^{\text{p}}$ and $E_{i,j}^{p}$, have an inverse effect on the grade since longer travel times and higher energy costs make a path less favourable path. Thus, the path grade combines these 3 factors.

4.2. Supervisory MPC-based Controller

Within the cooperative SaR architecture, an assumption is made that the supervisory control is centrally hosted on an external server that continuously gathers agent state and environment information. Based on this, it gives control commands to the agents. As stated before, the supervisory controller is meant to resolve what can be termed as an 'agent-environment' conflict, i.e, re-allocating SaR agents to regions of the environment that are suited to be searched by the specific agent. In this research, by definition an 'agent-environment' conflict occurs under two conditions: when for a UAV the mean value of $o(x,y)$ for all cells in its perception field is less than a certain threshold I_1 and when for a UGV the mean value of $t(x,y)$ for all cells in its perception field is greater than a certain threshold I_2. If either of the two conditions is true, the supervisory controller takes over and assigns globally optimal tasks to the SaR agents.

In this research, a model-predictive control (MPC) approach, is applied to the supervisory control role. This is motivated by the ability of MPC to provide optimal solutions for global objectives like total area coverage and victim detection over a certain prediction horizon while systematically incorporating constraints. A process model of the SaR environment is employed to predict the outcomes of candidate action sequences, over a prediction horizon $H_{i,j}^{p}$. The supervisory controller finds an optimal control sequence for all SaR agents, by solving an optimisation problem for a given cost function. The first step of this sequence is implemented, after which the horizon is shifted. The resulting optimal sequence comprises the set of optimal paths for all SaR agents to track. The controller subsequently executes the first action from this sequence, representing the immediate optimal step for all agents. In this way, the supervisory layer is optimization-based, and thus, distinct from the onboard heuristic FLC controllers of each agent. Under normal conditions, local heuristics adequately steer individual agents. However, when an agent–environment conflict emerges, the supervisory controller guides the agents to specific regions of the environment (with global optimality in consideration) and once all agents are in regions which are suited for their operational capacities, the control is handed over to the local controller.

The trade-off is the increased computational cost of solving an optimisation problem. However, according to the previous assumption of the supervisory layer being hosted on an external server, the increased computational burden can be afforded. Furthermore, faster convergence to an optimal solution can reduce the computation time and hence the associated cost. This can be achieved by providing the supervisory controller with the set of paths determined by each SaR agent's local controller as an initial 'warm start' solution [38].

4.2.1. MPC Objective Function

The supervisory controller maximises the following objective function:

$$J(\mathcal{P}) = w_1 \sum_{i=1,j=0|1}^{N} G_{i,j}^{p} + w_2 \sum_{(x,y)\in E} c(x,y,H_{i,j}^{p}) - w_3 \sum_{i=1,j=0|1}^{N} \sum_{(x,y)\in p_i} |A_{i,j} - o(x,y)| + |A_{i,j} - t(x,y)| \tag{15}$$

where $\mathcal{P}$ represents the set of paths of all N SaR agents and is the variable which is optimised in this MPC problem. The first term in the formulation represents the overall grade of all individual paths (refer to equation (14)). The second term represents the total predicted environmental scan certainty at the end of the prediction horizon $H_{i,j}^{p}$. Finally, the third term represents a factor which can be termed as a 'suitability sum'. This is based on the concept of 'task-resource' matching concept proposed by Fazal et al. [49], where a robot is dynamically assigned tasks based on the disparity between the quantity and types of resources it has and the specific resource requirements of the task. In the proposed formulation, $A_{i,j}$ is a constant which represents the agent (receives a value of $A_{i,1} = 1$ for UAVs and $A_{i,0} = 0$ for UGVs). The agent can be considered analogous to a 'resource'. While $o(x,y)$ and $t(x,y)$, can be considered analogous to a 'task', which requires a certain resource to be accomplished. The idea is to minimise the disparity between the resource requirement for a certain task and the available resources. This term models this disparity as the sum of difference between $A_{i,j}$ and $o(x,y)$ and $A_{i,j}$ and $t(x,y)$, for all cells (x,y) in the agent's path. For instance, consider a case where the agent is a UAV and there are two paths being compared: one which is high observability and high terrain index, while the other is the opposite with low observability and low terrain index. Here, the difference of $A_{i,1}$ with $o(x,y)$ and $t(x,y)$ will be higher in the second case, as the resource, i.e the 'UAV' is suited to the first task, i.e, 'scanning high observability, high terrain index cells'. Similarly, for the UGV the difference would be low for cells with low observability and low terrain index, indicating a low task-resource disparity. It will be high for high-observability, high terrain index cells. Overall, the system tries to minimise this term. In order to promote the agents to spread out over the environment, the objective function integrates both scan certainty and suitability sum terms. This is intended to yield a higher total scan certainty and optimised allocation of regions of interest to agents.

In contrast to the decentralised local controllers, the supervisory controller utilises the collective information from all agents, and communicates optimised paths to each SaR agent (illustrated in Figure 3). Through such bi-directional communication, the supervisory controller can continuously update and utilise a global representation of scan certainty $\mathcal{C}(\tau)$, terrain-index $\mathcal{T}(\tau)$ and observability $\mathcal{O}(\tau)$ maps of the environment. Further, it possesses a combined view of victims detected by merging individual victim maps $\mathcal{V}_i$, in the form of a global victim map $\mathcal{V}(\tau)$, which contains information of victims that have already been detected.

4.2.2. Optimisation Problem Formulation

The optimal control problem for the MPC is defined as follows:

$$\max_{\mathcal{P}} \quad J(\mathcal{P})$$

subject to:

$$\mathcal{P} = \{p_i, \dots ; \quad i = 1, \dots N| \quad p \in \mathbb{P}^{\mathrm{f}}\} \tag{14a}$$

$$\begin{aligned} p_i = \{(x^a_{1,i}, y^a_{1,i}), (x^a_{2,i}, y^a_{2,i}), \dots, (x^a_{n,i}, y^a_{n,i}) \mid \\ (x,y) \in E, (x,y) \notin \mathcal{W}(\tau)\} \end{aligned} \tag{14b}$$

$$\begin{aligned} (x^{\mathrm{vic}}_v, y^{\mathrm{vic}}_v) \notin \{p_Q \cap p_R \mid Q, R \in \{1, \dots, N\} \wedge Q \neq R\} \\ \forall v = 1, \dots, V \end{aligned} \tag{14c}$$

$$(x^a_{1,i}, y^a_{1,i}) = s_i(x_i, y_i, \tau) \quad \forall \quad i = 1, \dots, N \tag{14d}$$

where constraint (14a) ensures that the paths of all N SaR agents belong to the set of feasible paths $\mathbb{P}^{\mathrm{f}}$, meaning that each path p_i consists of n adjacent and consecutive cells. Constraint (14b) further limits paths to stay within the environment E and avoid cells by an obstacle (contained in $\mathcal{W}(\tau)$). The constraint (14c) prevents multiple agents from prioritising the same victim by ensuring that each victim's location (denoted with superscript vic) appears in at most one agent's path. This improves both search efficiency and area coverage. Lastly, constraint (14d) sets each agent's starting position at its current location $s_i(x_i, y_i, \tau)$, ensuring that path planning begins from the agent's actual position at the time of optimisation.

Figure 6 shows an overview of the process flow for the proposed control approach.

5. Case Study

In this section, the numerical simulations conducted to evaluate the performance of the proposed controller are detailed.

5.1. Compared Search Approaches

The hierarchical control structure proposed in this study is referred to as the 'cooperative' search approach, as the supervisory controller facilitates coordination among SaR agents. To evaluate its effectiveness, four alternative search strategies are considered for comparison.

In this study, the performance of these approaches is evaluated based on four key criteria: victim detection, area coverage, energy consumption, and an efficiency ratio that integrates these three factors. By analysing each criterion separately and in combination through the efficiency ratio, the effectiveness of all approaches in balancing mission objectives, which are often conflicting, is assessed. It is hypothesised that combining the local FLC controller, with the MPC controller in a supervisory role, will help achieve superior performance in victim detection and area coverage while maintaining reasonable energy efficiency.

Firstly, we consider a *selfish search approach*, where agents operate solely based on their local controllers, without coordination from a supervisory controller. This method serves to help study the impact of adding a supervisory control layer on the SaR system's performance. The other 3 approaches studied are a purely optimisation-based MPC controller, a heuristic controller and a random/exhaustive search approach. The pure-MPC controller has a similar structure as the supervisory MPC controller in the cooperative approach (described in Section 4.2.1) but without warm-start initialisation from local controllers. Instead, it solves the optimisation problem at each time step using a random initialisation, P_0. The heuristic method considered here is the *Ant Colony System (ACS) approach*. In this application of the ACS approach, the pheromone map is constructed in a manner inspired by de Koning's implementation in [38], using a weighted combination of scan certainty and environmental constraints, ensuring that agents prioritise both exploration and environmental feasibility in their movement decisions. For each agent, the pheromone value of a cell is determined differently. On one hand, UGVs prioritise cells with low scan certainty while favouring areas with low terrain difficulty. This is achieved by inverse scaling of the scan certainty map $\mathcal{C}(\tau)$ by the terrain index map $\mathcal{T}(\tau)$, encouraging movement through less obstructed regions. On the other hand, UAVs prioritise cells with low scan certainty while favouring regions with high observability. The pheromone strength here is adjusted by proportional scaling of the scan certainty by the observability map $\mathcal{O}(\tau)$. Finally, we also consider a *random or exhaustive search approach*. This serves as a baseline benchmark for comparison. Such an approach may be beneficial in cases where there is no initial information available about the environment. While such an approach may be useful in cases where no prior information about the environment is available, it lacks the ability to execute a structured search strategy, limiting its overall effectiveness.

5.2. Randomised Environment Simulations

The cooperative search approach and comparative methods were evaluated across 20 simulation scenarios, each running for a duration of $\tau^{\max}$ time steps. To control variability, terrain conditions, observability levels, victim locations, initial agent positions, and obstacle placements were generated using a seed value. A summary of key environmental characteristics is presented in Table 3. The physical environmental conditions of the environment were varied by dividing it into eight equally sized regions, with each region assigned to one of four possible conditions: high observability–high terrain difficulty, high observability–low terrain difficulty, low observability–low terrain difficulty, and low observability–high terrain difficulty. These regions were randomly placed in each simulation instance to ensure varied test conditions. Table 3 provides a comprehensive overview of several parameters used in the control framework. The experiments were conducted with $N = 2$ SaR agents, comprising of one UAV (Agent 1) and one UGV (Agent 2). As discussed in Section 3.2, the agents exhibit heterogeneous sensory capabilities, differing in sensory perception radii $r^{\mathrm{p}}_{i,j}$, perceptual uncertainty reduction rates $\eta_{i,j}$, and occlusion sensitivity factor, $\alpha_{i,j}$. Both agents begin each simulation without prior knowledge of the environment, meaning the scan certainty map is initialised as completely uncertain, i.e. $c(x, y, \tau_0) = 0 \quad \forall (x, y) \in E$.

As discussed in Section 4, the controller aims to optimise for area coverage, victim detection, and energy consumption. These objectives are often conflicting. To compare the performance of different search strategies, multiple performance indicators are analysed. For area coverage, two metrics are taken into account. First, the total achieved scan certainty of the environment over time is considered. We define it as:

$$S(\tau) = \sum_{(x,y)\in E} c(x, y, \tau) \tag{16}$$

This can be seen in Figure 7. Secondly, the rise time of the total scan certainty, measuring the number of time steps required to reach 15%, 25%, 30% and 35% of the total achievable environmental scan certainty, is considered. The rise times are obtained from the plot

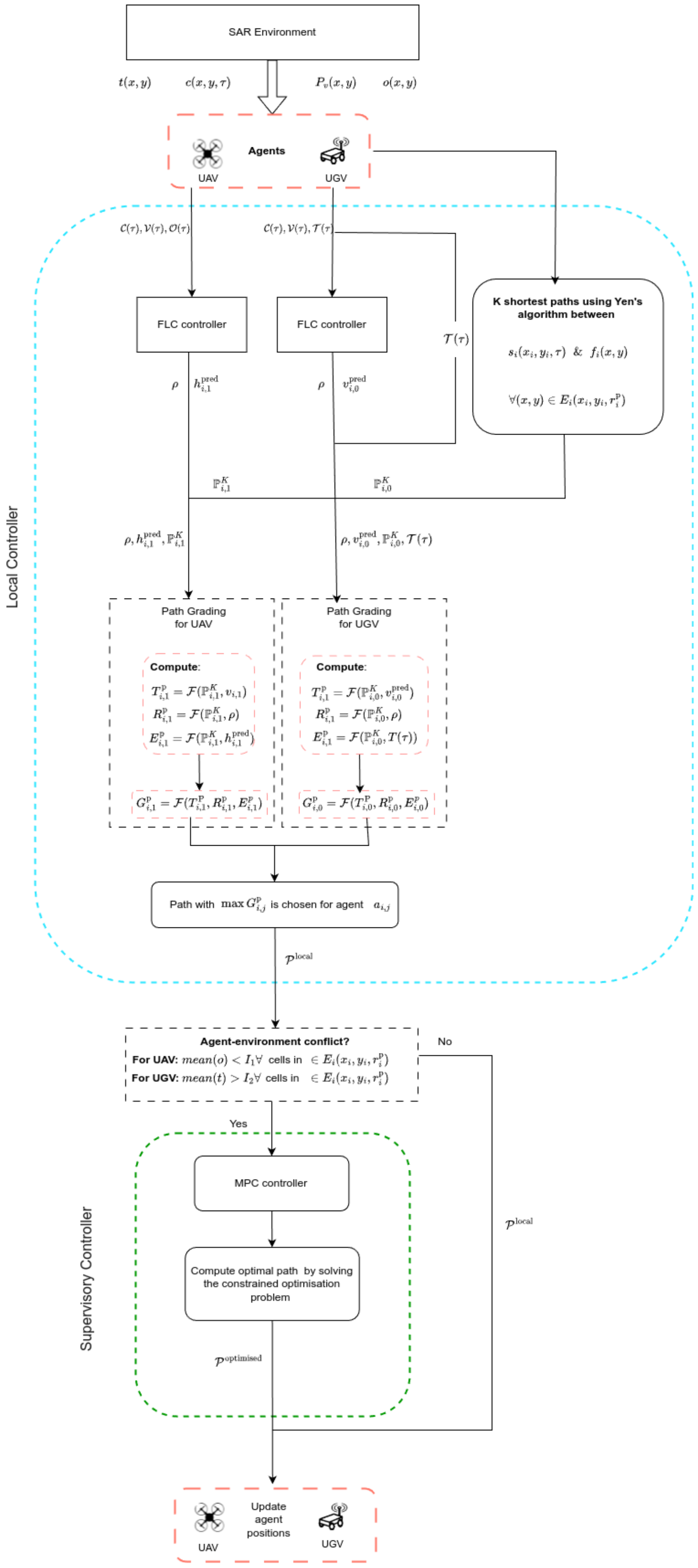

SAR Environment
$t(x,y)$
$c(x,y,\tau)$
$P_v(x,y)$
$o(x,y)$
Agents
UAV
UGV
Local Controller
$\mathcal{C}(\tau),\mathcal{V}(\tau),\mathcal{O}(\tau)$
$\mathcal{C}(\tau),\mathcal{V}(\tau),\mathcal{T}(\tau)$
FLC controller
FLC controller
$\mathcal{T}(\tau)$
K shortest paths using Yen's algorithm between
$s_i(x_i,y_i,\tau)$ & $f_i(x,y)$
$\forall(x,y)\in E_i(x_i,y_i,r_i^{\mathrm{p}})$
ρ
$h_{i,1}^{\mathrm{pred}}$
ρ
$v_{i,0}^{\mathrm{pred}}$
$\mathbb{P}_{i,1}^K$
$\mathbb{P}_{i,0}^K$
$\rho, h_{i,1}^{\mathrm{pred}}, \mathbb{P}_{i,1}^K$
$\rho, v_{i,0}^{\mathrm{pred}}, \mathbb{P}_{i,0}^K, \mathcal{T}(\tau)$
Path Grading for UAV
Compute:
$T_{i,1}^{\mathrm{p}}=\mathcal{F}(\mathbb{P}_{i,1}^K, v_{i,1})$
$R_{i,1}^{\mathrm{p}}=\mathcal{F}(\mathbb{P}_{i,1}^K, \rho)$
$E_{i,1}^{\mathrm{p}}=\mathcal{F}(\mathbb{P}_{i,1}^K, h_{i,1}^{\mathrm{pred}})$
$G_{i,1}^{\mathrm{p}}=\mathcal{F}(T_{i,1}^{\mathrm{P}}, R_{i,1}^{\mathrm{p}}, E_{i,1}^{p})$
Path Grading for UGV
Compute:
$T_{i,1}^{\mathrm{p}}=\mathcal{F}(\mathbb{P}_{i,0}^K, v_{i,0}^{\mathrm{pred}})$
$R_{i,1}^{\mathrm{p}}=\mathcal{F}(\mathbb{P}_{i,0}^K, \rho)$
$E_{i,1}^{\mathrm{p}}=\mathcal{F}(\mathbb{P}_{i,0}^K, T(\tau))$
$G_{i,0}^{\mathrm{p}}=\mathcal{F}(T_{i,0}^{\mathrm{P}}, R_{i,0}^{\mathrm{p}}, E_{i,0}^{p})$
Path with $\max G_{i,j}^{\mathrm{p}}$ is chosen for agent $a_{i,j}$
$\mathcal{P}^{\mathrm{local}}$
Agent-environment conflict?
For UAV: $mean(o) < I_1 \forall$ cells in $\in E_i(x_i,y_i,r_i^{\mathrm{p}})$
For UGV: $mean(t) > I_2 \forall$ cells in $\in E_i(x_i,y_i,r_i^{\mathrm{p}})$
No
Yes
Supervisory Controller
MPC controller
Compute optimal path by solving the constrained optimisation problem
$\mathcal{P}^{\mathrm{optimised}}$
$\mathcal{P}^{\mathrm{local}}$
UAV
Update agent positions
UGV

Figure 6. Process flow overview for proposed control approach; here $\rho \rightarrow \rho(x,y,\tau), h_{i,1}^{\mathrm{pred}} \rightarrow h_{i,1}^{\mathrm{pred}}(x,y), v_{i,1}^{\mathrm{pred}} \rightarrow v_{i,1}^{\mathrm{pred}}(x,y)\ \forall(x,y) \in E_i(x_i,y_i,r_i^{\mathrm{p}})$; $\mathcal{P}^{\mathrm{local}}$ and $\mathcal{P}^{\mathrm{optimised}}$ represent the set of paths chosen by the local and supervisory controller, respectively

Parameter	Value
(L_y,L_x)	(24,24)
V	20
p_s	0.6 or 60%
K	3
γ	0.6
Γ	0.5
Δ	5
Θ	3
I_1	0.6
I_2	0.5
w_1	2
w_2	1
w_3	1.5
$\tau^{\max}$	150

Table 3. Simulation Modelling parameters

in Figure 7 and are presented in Table 4. Together, these indicators evaluate both the absolute area coverage and its time evolution for each search approach.

	$\tau_s \rightarrow 15\%$	$\tau_s \rightarrow 25\%$	$\tau_s \rightarrow 30\%$	$\tau_s \rightarrow 35\%$
Cooperative	43	82	117	-
Selfish	52	146	-	-
Pure-MPC	47	-	-	-
ACS	39	76	97	121
Exhaustive	91	-	-	-

Table 4. Rise time for achieving different levels of total scan certainty in a randomised environment

The victim detection efficiency is evaluated by analysing the (i) total number of victims found at the end of the simulation and (ii) the time taken to find each victim. These parameters are shown in Figure 8 using box-plots. Further, the energy efficiency is assessed using a cumulative plot of energy consumption over time, as shown in Figure 9. Finally, the approaches are compared on the basis of an Efficiency ratio, which integrates all three objectives—victim detection, area coverage, and energy consumption—into a single metric. This ratio evaluates how effectively each approach balances these competing factors to maximise overall performance. It is computed as:

$$\zeta(\tau) = \frac{\Delta \cdot v(\tau) + \Delta \cdot s(\tau)}{\Theta \cdot e(\tau)} \tag{17}$$

where $v(\tau)$ represents the number of victims found at time step τ. Similarly, $s(\tau)$ and $e(\tau)$ represent the improvement in scan certainty and the energy consumed at time step τ relative to the previous state at time step $\tau - 1$. This metric, thus, quantifies the energy cost per unit improvement in area coverage and victim detection. Figure 10 shows the efficiency ratio over time for the different search approaches.

5.3. Special Simulation Cases

In addition to simulations in randomised environments, the cooperative search approach and the selfish search approach are evaluated in specially designed, small-scale environments. These targeted scenarios demonstrate the advantages of coordination over purely local decision-making. Unless specified otherwise, the modelling parameters remain consistent with Table 3. Also, Table 5 provides the specific agent parameters for each test case.

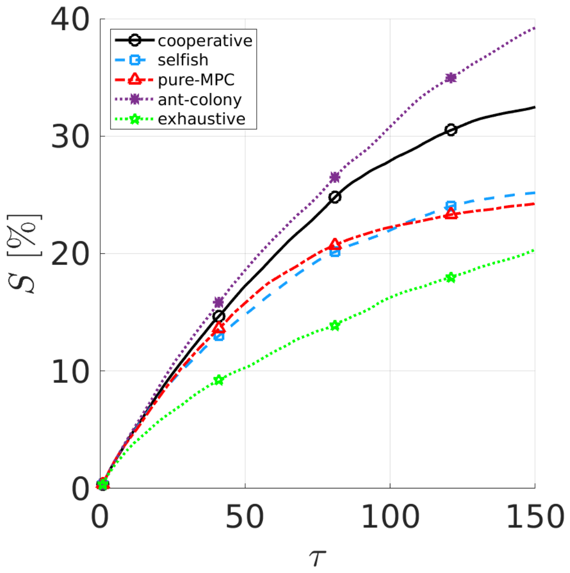


Figure 7. Total scan certainty achieved by each search strategy in a randomized environment, expressed as a percentage of the maximum possible scan certainty across the entire environment

		(x_0, y_0)	$r^{\mathrm{p}}_{i,j}$	$\eta_{i,j}$	$\alpha_{i,j}$
Case 1	Agent 1	(9,8)	6	0.15	0.8
	Agent 2	(9,10)	3	0.15	0.2
Case 2	Agent 1	(15,8)	6	0.15	0.8
	Agent 2	(15.10)	3	0.15	0.2
Case 3	Agent 1	(9,2)	6	0.15	0.8
	Agent 2	(9,3)	2	0.15	0.2

Table 5. Agent parameters for special simulation cases

5.3.1. Case 1

The first simulation case features an environment with moderate observability and terrain difficulty. Within this space, two distinct sub-regions are defined: Region A, characterised by higher observability, and Region B, which has lower observability than the rest of the environment. Both regions exhibit lower scan certainty compared to the surrounding area. At the initial time step τ_0, scan certainty is distributed as:

$$c(x, y, \tau_0) = \begin{cases} 0 & \forall (x, y) \in (E_A \cup E_B) \\ 0.95 & \text{otherwise} \end{cases}$$

and victims are distributed as shown in the Table 6.

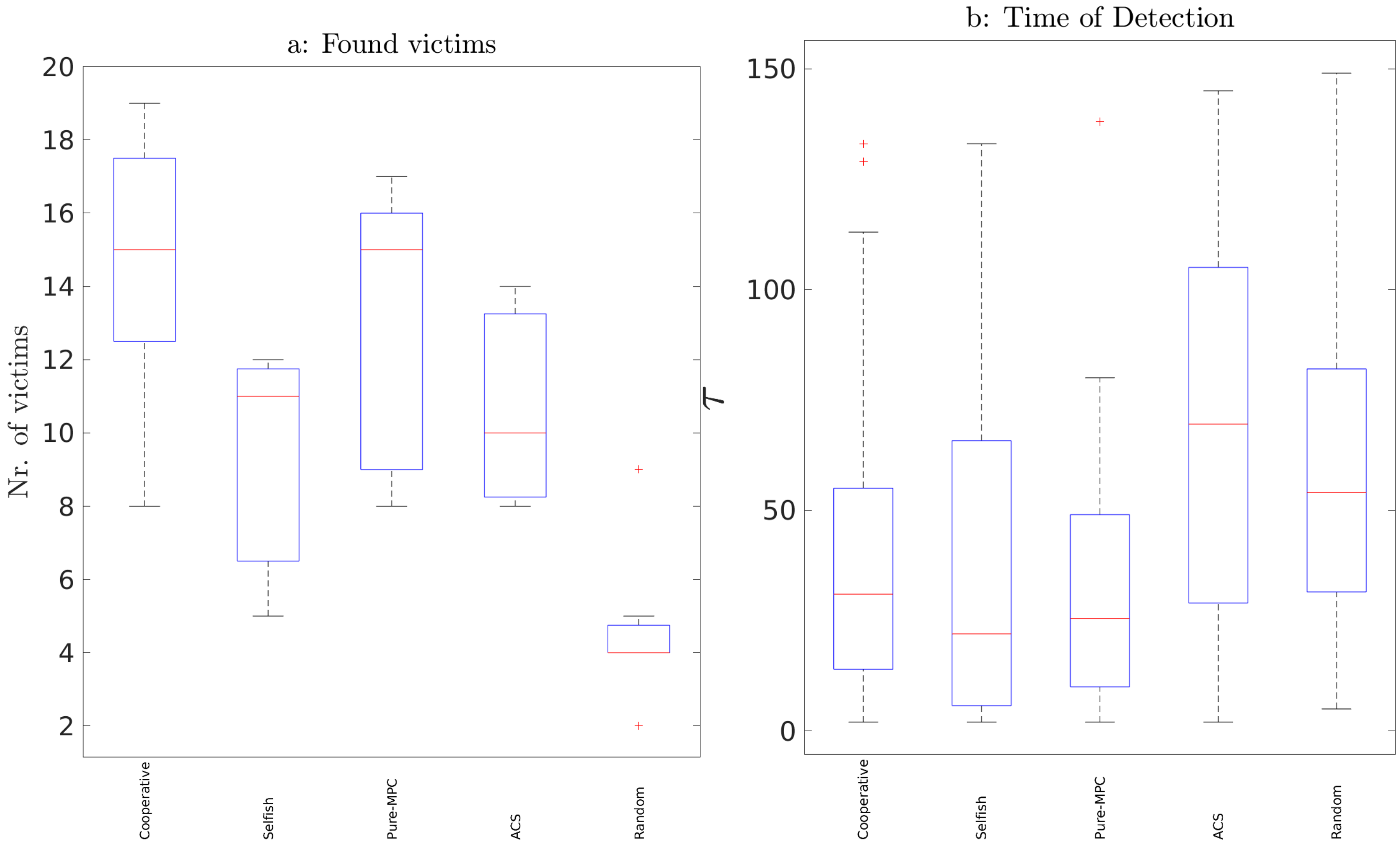
Victim	Position
v_1	$(2,3)$
v_2	$(5,5)$
v_3	$(6,5)$
v_4	$(6,6)$
v_5	$(11,1)$
v_6	$(13,2)$
v_7	$(13,3)$
v_8	$(12,4)$

Table 6. Victim locations in simulation Case 1

This simulation case is illustrated in Figure 12. The path executed by the agents over $\tau^{\max} = 25$ time-steps can be seen in Figures 15 and 16. Additionally, key performance metrics, including scan certainty progression (ΔS vs τ), victim detection efficiency, energy consumption (ΔE vs τ) and efficiency ratio (ζ vs τ) are presented in Figures 21 - 23.

5.3.2. Case 2

In the second case, we consider an environment with low observability and high terrain difficulty. Within this setting, two sub-regions are defined: Region A, which has high observability and low terrain



(a) Number of victims found

(b) Time taken to find victims

Figure 8. Victim Detection Efficiency for search approaches in randomised simulation environment

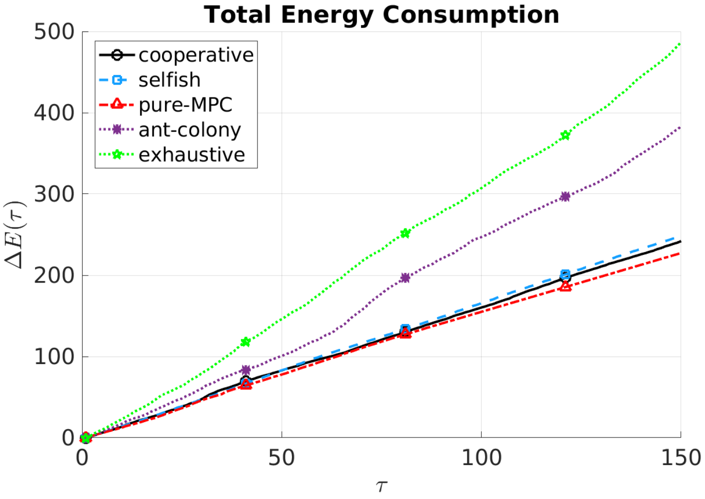


Figure 9. The total energy consumption for each search approach in the randomised simulation environment

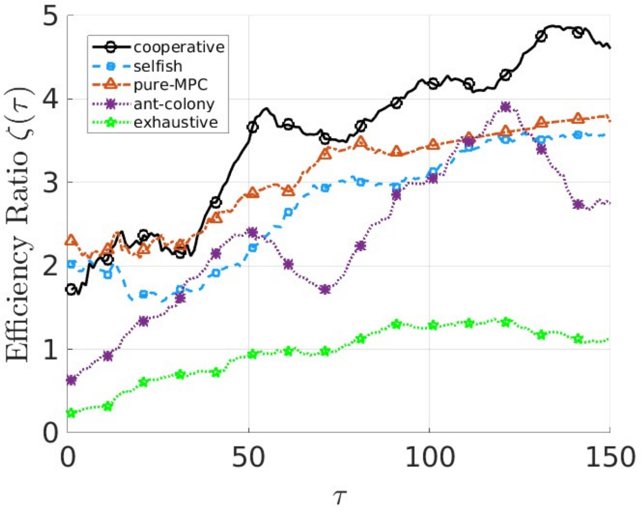


Figure 10. The efficiency ratio for each search approach in the randomised simulation environment

difficulty, and Region B, characterised by low observability and high terrain difficulty. Both regions exhibit lower scan certainty than the surrounding environment. However, Region A has the lowest scan certainty, meaning it is almost entirely unexplored, while Region B has partial scan certainty, indicating that some prior knowledge of the region is available. This scenario does not contain any victims. At the initial time step τ_0, scan certainty is distributed as:

$$c(x, y, \tau_0) = \begin{cases} 0.2 & \forall (x, y) \in E_A \\ 0.7 & \forall (x, y) \in E_B \\ 0.95 & \text{otherwise} \end{cases}$$

A visual representation of this scenario is provided in Figure 13. The path executed by the agents over $\tau^{\max} = 25$ time-steps can be seen in Figures 17 and 19. Additionally, the key performance metrics, including scan certainty progression (ΔS vs τ), victim detection efficiency, energy consumption (ΔE vs τ) and efficiency ratio (ζ vs τ) are presented in Figures 25 - 27.

5.3.3. *Case 3*

In the third case, we consider an environment with moderate observability and low terrain difficulty. Within this space, Region A is defined by lower observability and higher terrain difficulty. Nested

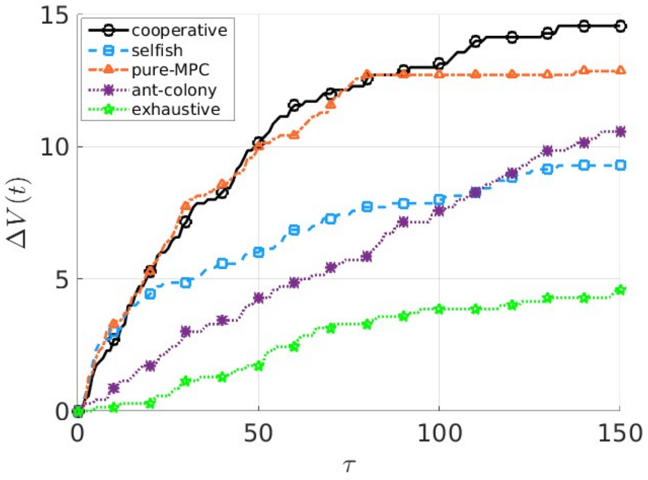


Figure 11. Average number of victims detected at each time-step

within Region A, Region B presents a more challenging task for both SaR agents, with very low observability and difficult terrain. This region is surrounded by dense obstacles (e.g., walls), making it inaccessible except through a single opening. Both regions contain victims, but Region B has a higher victim density. Additionally, scan certainty is significantly lower in Region B, while partial scan certainty exists for Region A. At the initial time step τ_0, scan certainty is distributed as:

$$c(x, y, \tau_0) = \begin{cases} 0.7 & \forall (x, y) \in E_A \\ 0.2 & \forall (x, y) \in E_B \\ 0.95 & \text{otherwise} \end{cases}$$

The victim locations in the environment can be seen in Table 7.

Victim	Position
v_1	$(10, 8)$
v_2	$(11, 7)$
v_3	$(12, 5)$
v_4	$(11, 5)$
v_5	$(10, 5)$
v_6	$(13, 2)$
v_7	$(10, 11)$
v_8	$(11, 6)$

Table 7. Victim locations in simulation Case 3

This simulation case is illustrated in Figure 14. The path executed by the agents over $\tau^{\max} = 25$ time-steps can be seen in Figures 19 and 20. Additionally, key performance metrics, including scan certainty progression (ΔS vs τ), victim detection efficiency, energy consumption (ΔE vs τ) and efficiency ratio (ζ vs τ) are presented in Figures 28 - 31.

6. Results and Discussion

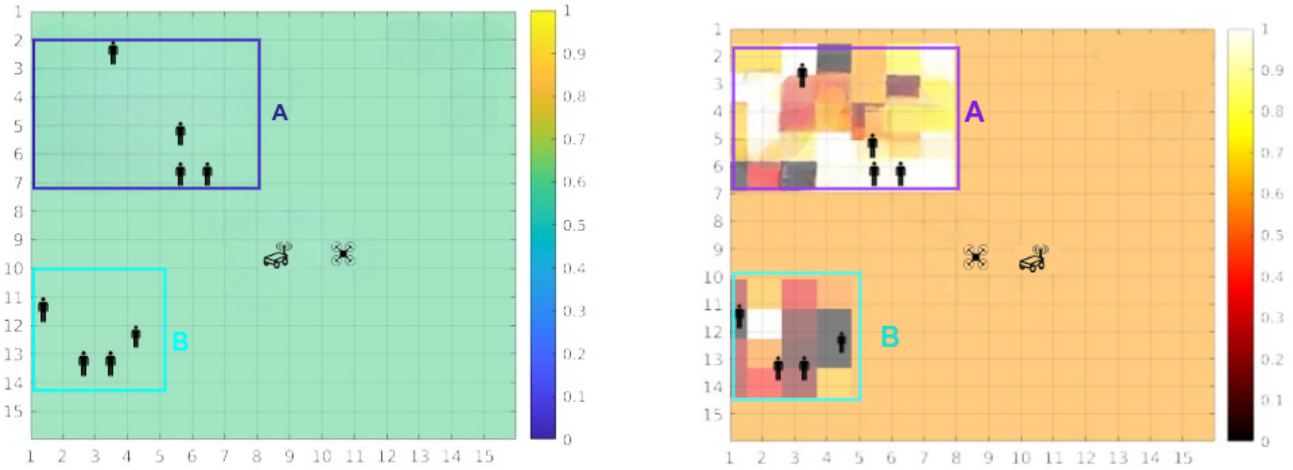
6.1. Randomised Environment Simulations

First, we will discuss the performance in terms of area coverage. In Figure 7, it can be seen that the cooperative approach achieves a higher scan certainty than the cooperative approach achieves higher scan certainty compared to the selfish, pure-MPC, and exhaustive search approaches. The final scan certainty in the cooperative case is 29% higher than the selfish approach and 33% higher than the pure-MPC method. Additionally, the cooperative approach demonstrates lower rise times, as detailed in Table 4. The ACS approach, however, outperforms all methods in scan certainty, achieving 20.8% higher coverage than the cooperative approach. It also exhibits the fastest rise times, reaching 15%, 25%, 30%, and 35% scan certainty quicker than the other strategies. This is likely due to the fact that its objective function is directly designed to maximise exploration by prioritising cells with low scan certainty while incorporating environmental feasibility, but as a scaling factor and not a primary objective. Unlike the cooperative, selfish, and pure-MPC approaches, which balance multiple objectives such as victim detection and energy efficiency, ACS focuses primarily on increasing scan coverage, leading to a more aggressive exploration strategy. The cooperative approach, for instance, distributes effort between finding victims, optimising coverage, and energy consumption, leading to more balanced movement but slightly lower scan certainty gains compared to ACS.

The victim detection performance of each method is evaluated using two key indicators: central tendency and variability in detection outcomes. The median number of victims detected represents the general performance, while the interquartile range (IQR) quantifies the variability in detection results. As mentioned in Table 3, the total number of victims in the random simulation is 20. The cooperative approach achieves the highest median victim detection, with 15 victims found, meaning that in at least half of the simulations, it successfully detects more than 75% of the victims. The pure-MPC approach achieves the same median, but its IQR is 7, compared to 5 for the cooperative approach, indicating greater variability in its performance. While both methods detect the same median number of victims, the cooperative approach is more consistent. The selfish approach detects a median of 11 victims, which is 36% lower than both the cooperative and pure-MPC methods. The ACS approach fell further behind, detecting a median of only 10 victims. This is expected, as the ACS method does not explicitly search for victims. Thus, any victims found are merely a by-product of its exploration strategy rather than a direct search effort. The median detection time for the cooperative approach is 31 time steps, which is higher than the selfish (22 time steps) and pure-MPC (25.5 time steps) approaches. However, this is because the cooperative method finds more victims overall, many of which are detected later in the process. The supervisory controller ensures agents are strategically dispersed across the environment, leading to more thorough coverage over time. The selfish approach, on the other hand, detects fewer victims overall but finds half of them within the first 22 time steps. Beyond this point, it struggles to reach victims in distant areas, leading to both a lower overall victim count and a wider spread of detection times. This is reflected in its IQR of 60, compared to 41 for the cooperative approach. To further understand this, the cumulative victim detection plot (which shows the addition to victim count at each time-step) can be seen in Figure 11.

It can be seen that the cooperative approach continues to detect more victims in the latter time steps, in comparison to the selfish approach, which adds fewer victims to its count after the initial time steps. The exhaustive method, as expected, performs the worst both in terms of victim count and time of detection.

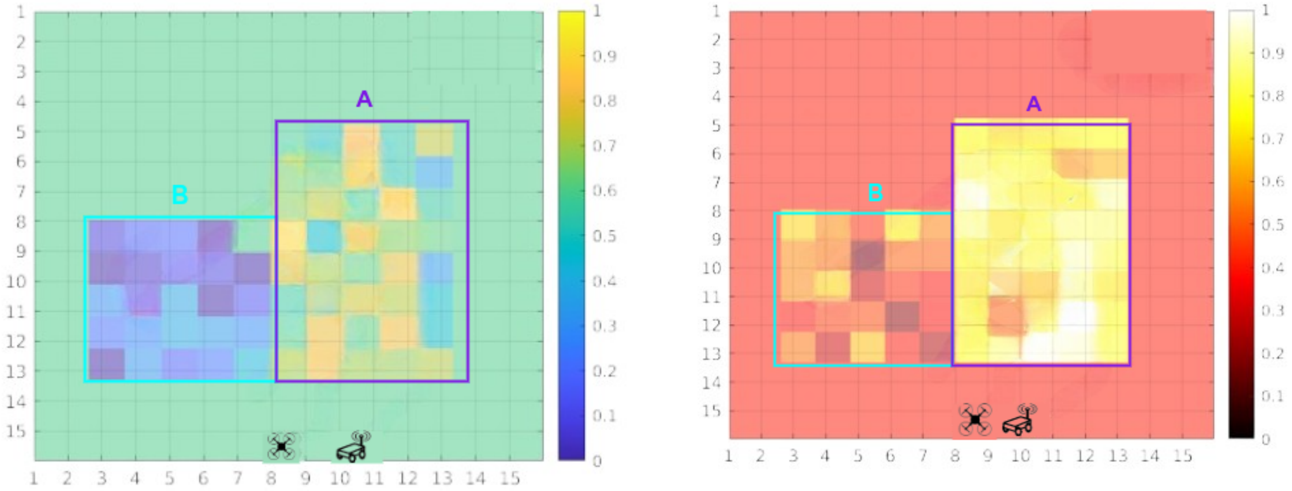
Further, the energy consumption of each approach is compared using the energy plot in Figure 9, which shows that the exhaustive approach is the least energy-efficient, followed by ACS. The cooperative approach consumes 3% less energy than the selfish approach and 7% more energy than the pure-MPC approach. While the difference in energy consumption is relatively small, it is important to note that the cooperative method outperforms the other approaches in both area coverage (except the ACS approach) and victim detection, achieving these results without a significant increase in energy usage. To further understand this, we consider



(a) Terrain Map

(b) Observability Map

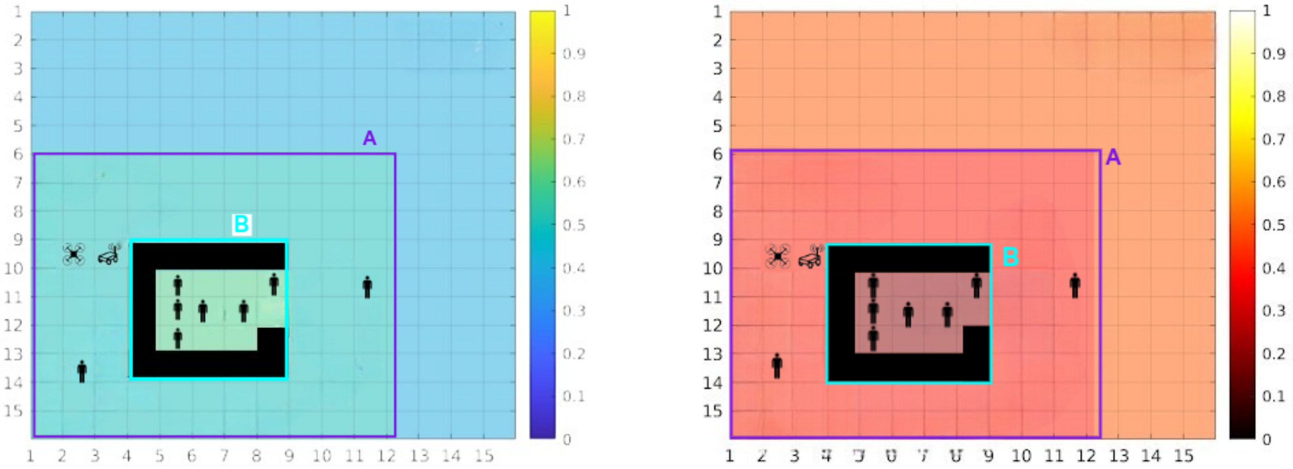
Figure 12. Simulation environment for **Case 1**



(a) Terrain Map

(b) Observability Map

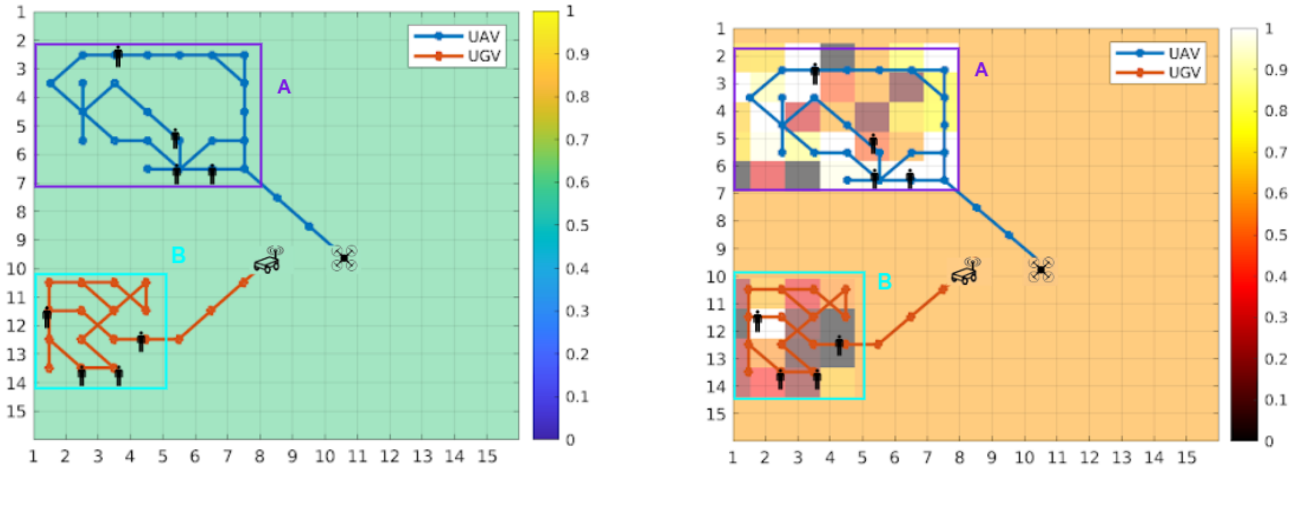
Figure 13. Simulation environment for **Case 2**



(a) Terrain Map

(b) Observability Map

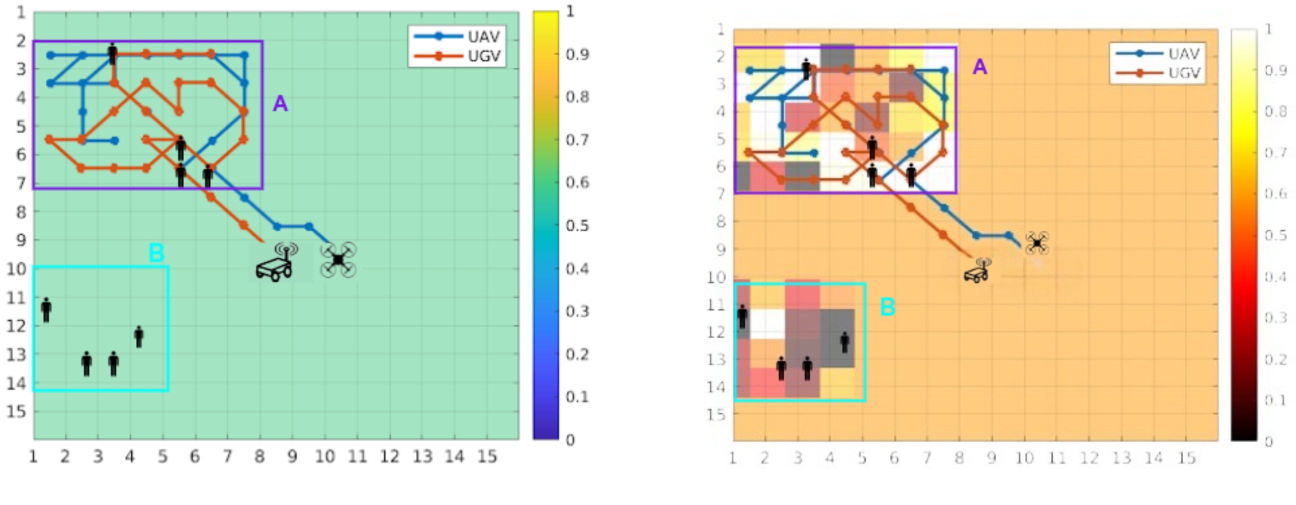
Figure 14. Simulation environment for **Case 3**



(a) Terrain Map

(b) Observability Map

Figure 15. Paths executed by the agents in the cooperative approach for **Case 1**



(a) Terrain Map

(b) Observability Map

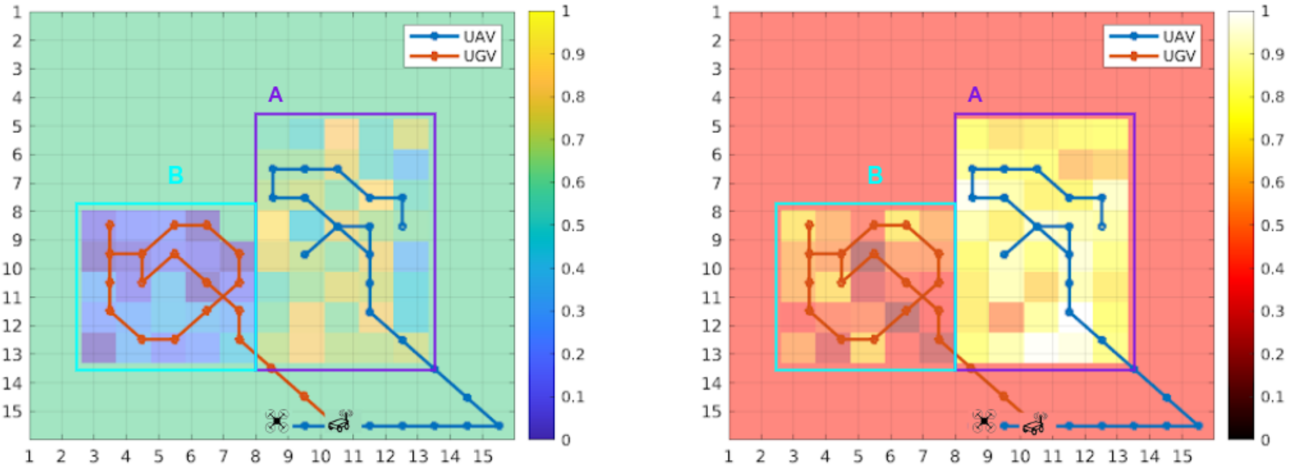
Figure 16. Paths executed by the agents in the selfish approach for **Case 1**

the Efficiency ratio plots for each method. Figure 10 shows that the cooperative approach maintains a higher efficiency ratio for approximately 76% of the simulation time, meaning it achieves better mission outcomes per unit of energy expended for the majority of time steps.

6.2. Special Simulation Cases

In this section, the performance of the search approaches in the special simulation cases is evaluated. First, we consider **Case 1**. As described in Section 5.3.1, the environment in Case 1 consists of two key regions of interest (Region A and Region B), both of which have low scan certainty and contain victims to be detected. The entire environment, including these regions, is characterised by high terrain difficulty, making it sub-optimal for the UGV. Region A has high observability, making it ideal for UAV scanning, while Region B has low observability, making it less suited for UAV operations. The optimal strategy is for the UAV to prioritise Region A, where it can scan effectively, while the UGV takes responsibility for scanning Region B, even though it remains a challenging task. This is a globally efficient strategy, which maximises coverage by distributing the workload between agents.

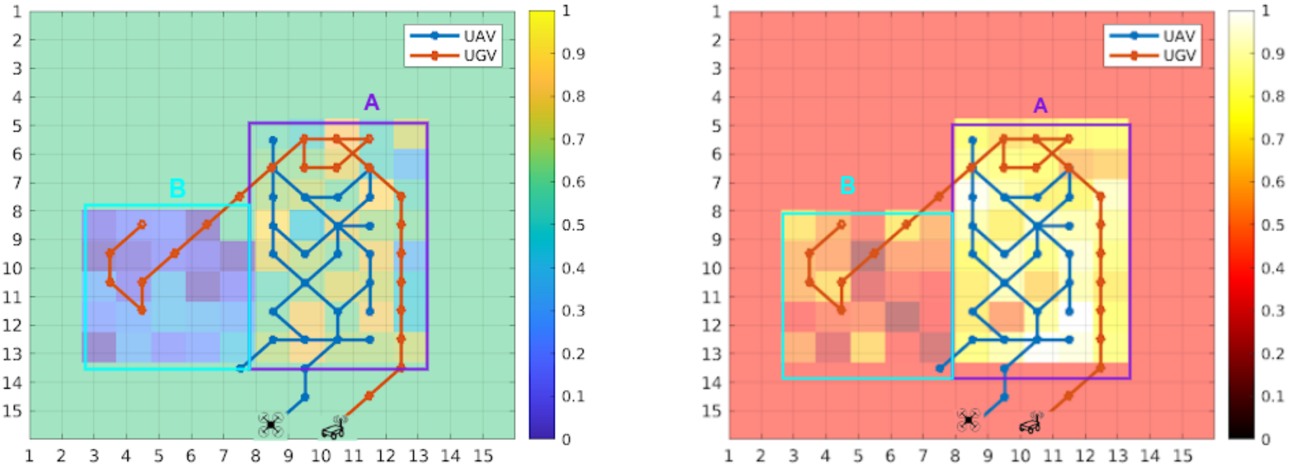
In the selfish scenario, it is observed that both the UGV and UAV approach and scan the same region (Region A). Here, the UAV naturally decides to scan the high-priority, high-observability region. For the UGV, both Regions A and B have similar visitation priority and terrain difficulty, making its choice ambiguous. In Figure 16, it can be observed that the UGV follows a selfish strategy, choosing to scan Region A, leading to both agents operating in the same area. This results in inefficient resource allocation, as Region B remains unscanned. In contrast, the cooperative approach, as shown in Figure 15, demonstrates improved task distribution. Here, the UAV is assigned to Region A, while the UGV simultaneously scans Region B, ensuring more effective coverage and better utilisation of agent capabilities. The impact of search behaviour differences on performance can be evaluated by analysing the previously mentioned performance criteria. In terms of area coverage, the cooperative approach manages to achieve a final scan certainty which is 18% higher than the selfish approach. This can be seeing in Figure 21. Figure 24 illustrates victim detection performance across both approaches. The cooperative method detects more victims overall, with a median of 8, meaning that in 50% of the simulation runs, all victims are successfully found. In contrast, the selfish approach has a median of 4.5 victims detected, indicating that in half of the



(a) Terrain Map

(b) Observability Map

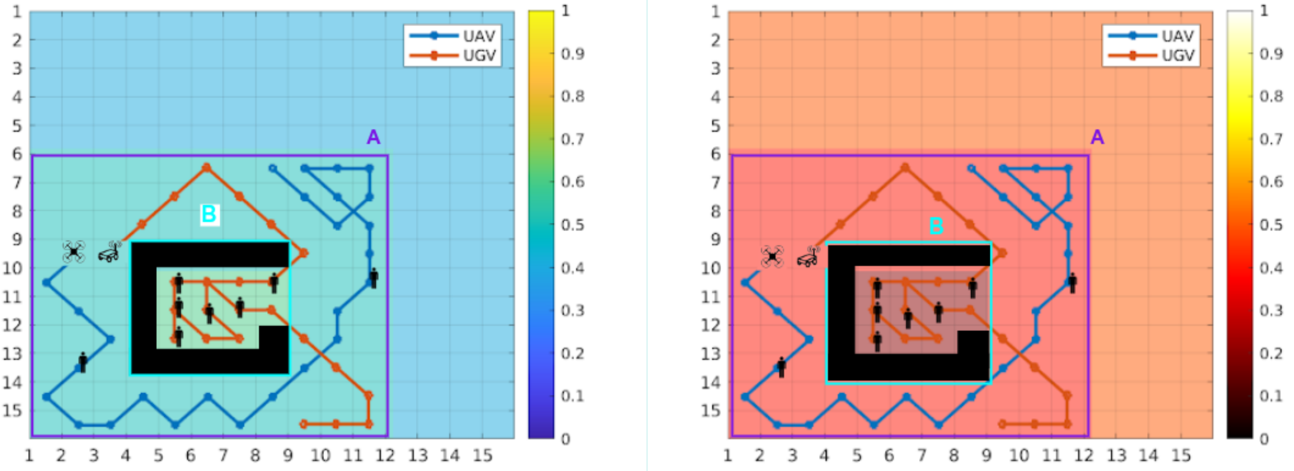
Figure 17. Paths executed by the agents in the cooperative approach for **Case 2**



(a) Terrain Map

(b) Observability Map

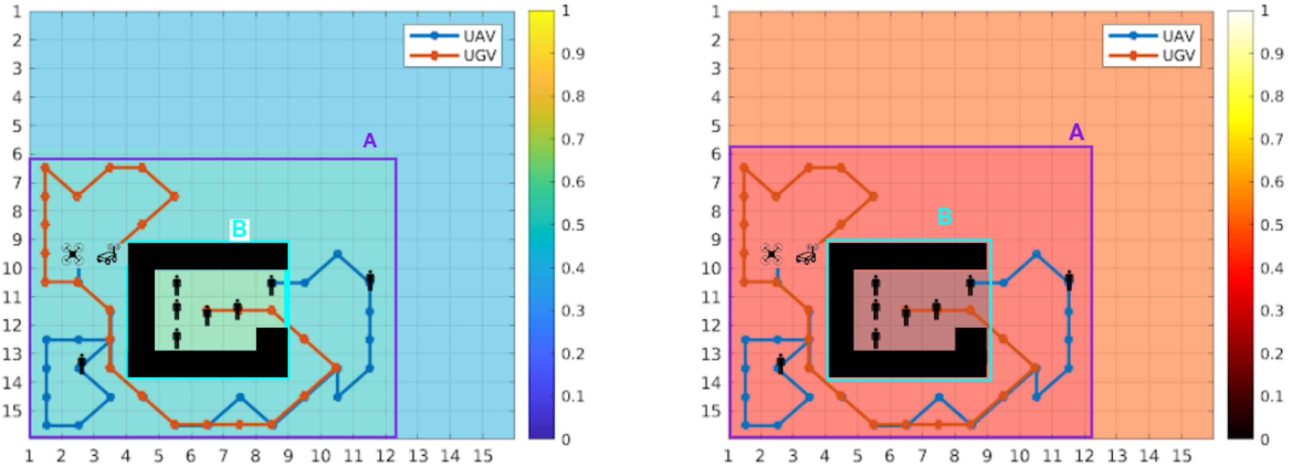
Figure 18. Paths executed by the agents in the selfish approach for **Case 2**



(a) Terrain Map

(b) Observability Map

Figure 19. Paths executed by the agents in the cooperative approach for **Case 3**



(a) Terrain Map

(b) Observability Map

Figure 20. Paths executed by the agents in the selfish approach for **Case 3**

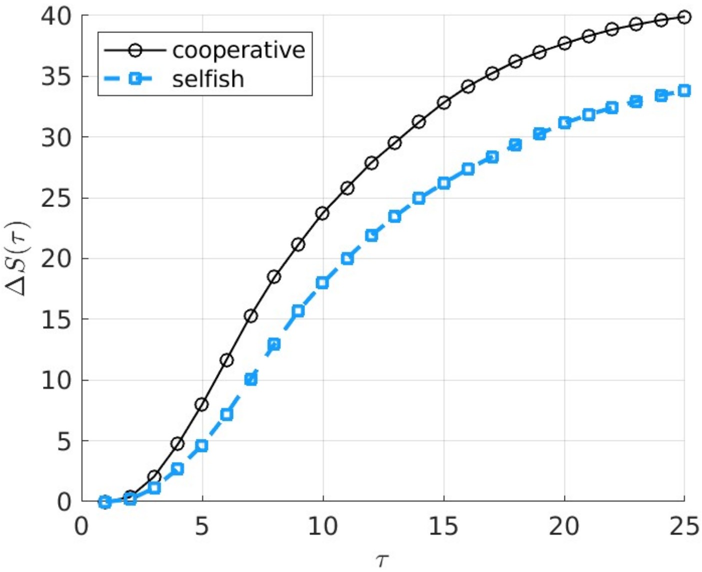


Figure 21. The total scan certainty for **Cooperative** and **Selfish** approaches in Case 1

cases, agents fail to locate victims outside of Region A. The median detection time is slightly lower for the selfish approach (9 time steps) compared to 10 for the cooperative method. This is likely because, in the selfish case, both agents focus on the same region (Region A), leading to faster victim detection within that area. However, this approach neglects Region B, reducing the total number of victims found. On the other hand, the cooperative method distributes agents across the environment, which results in higher overall victim detection at the cost of a slightly longer detection time due to independent scanning efforts.

In terms of energy consumption, both methods show comparable performance, with the cooperative approach resulting in a final energy consumption that is only 3% less than in the selfish case. This is expected as terrain conditions remain constant across both cases, and the UGV traverses a comparable number of steps regardless of the specific path taken. Likewise, in both approaches, the UAV eventually moves into the high-observability region (Region A)—either independently in the selfish approach or guided by the supervisory controller in the cooperative approach. Consequently, energy consumption remains largely unchanged, as shown in Figure 22.

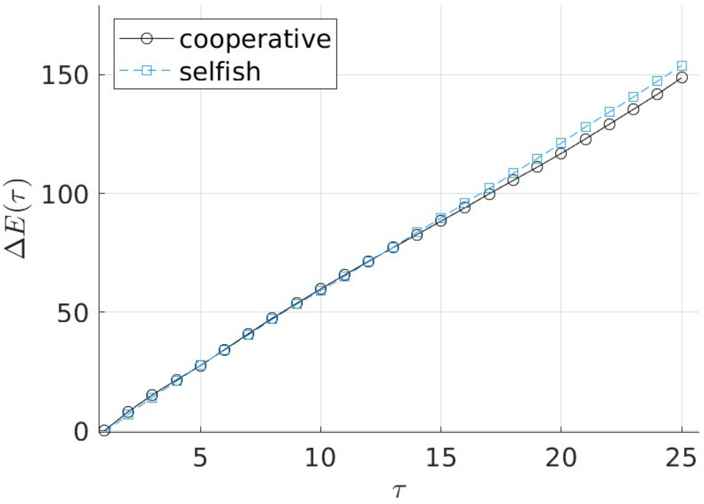


Figure 22. Comparison of Energy Consumption for **Cooperative** and **Selfish** approaches in Case 1

Finally, the efficiency ratio plot in Figure 23 indicates that the efficiency ratio for the cooperative approach maintains a higher efficiency ratio at all times. This shows that while both approaches consume nearly the same amount of energy, the cooperative method achieves significantly better area coverage and detects more victims overall.

Next, we consider **Case 2**. As mentioned in section 5.3.2, there exist two regions of interest: Region A, which is suited for exploration by the UAV, and Region B, which is optimal for exploration by the UGV. In the selfish approach, both agents initially prioritise Region A because it has lower scan certainty compared to Region B. The local controller prioritises scan certainty improvement over energy efficiency, though energy cost is still considered with a lower weight in the optimisation function. Consequently, the UGV selects Region A to maximise its immediate payoff. As the scan certainty in Region A improves over time, Region B becomes a higher-priority target, eventually prompting the UGV to shift its focus toward it. However, this delayed decision results in Region B being scanned at later time steps, reducing the overall efficiency of the search. This behaviour is illustrated in Figure 18. In contrast to this, in the cooperative

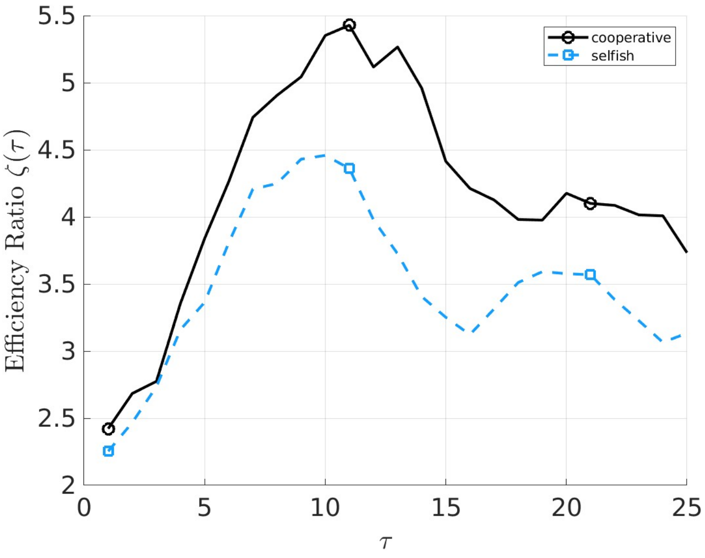


Figure 23. Comparison of Efficiency Ratio for **Cooperative** and **Selfish** approaches in Case 1

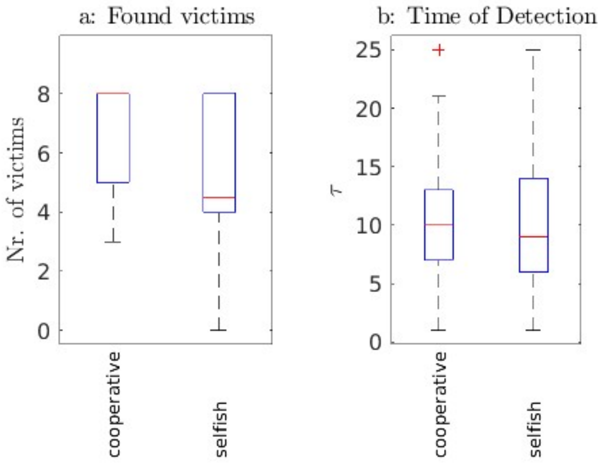


Figure 24. Comparison of Victim Detection Efficiency for **Cooperative** and **Selfish** approaches in Case 1

approach, the supervisory controller distributes the agents, assigning the UAV to Region A, where high observability conditions make it more effective, while the UGV is directed to Region B, which has lower observability but is easier to traverse. This allocation ensures that both regions are scanned efficiently from the beginning. This behaviour can be seen in Figure 17. In terms of area coverage, it outperforms the selfish approach, achieving a final scan certainty that is 24% higher, as shown in Figure 25.

The energy consumption for both search approaches can be seen in Figure 26. The cooperative approach consumes about 13.5% less energy in comparison to the selfish approach. This is expected since the supervisory controller directs the UGV to Region B, which has a lower terrain index, and operating here is less energy-intensive for the UGV. Further, Figure 27 shows that the efficiency ratio for the cooperative approach remains higher than the selfish approach throughout most time-steps (about 88% of the time steps). This shows that the cooperative approach consumes less energy while achieving significantly better area coverage.

Finally, we have a look at **Case 3**. As discussed in section 5.3.3, the entire environment presents sub-optimal conditions for both agents.

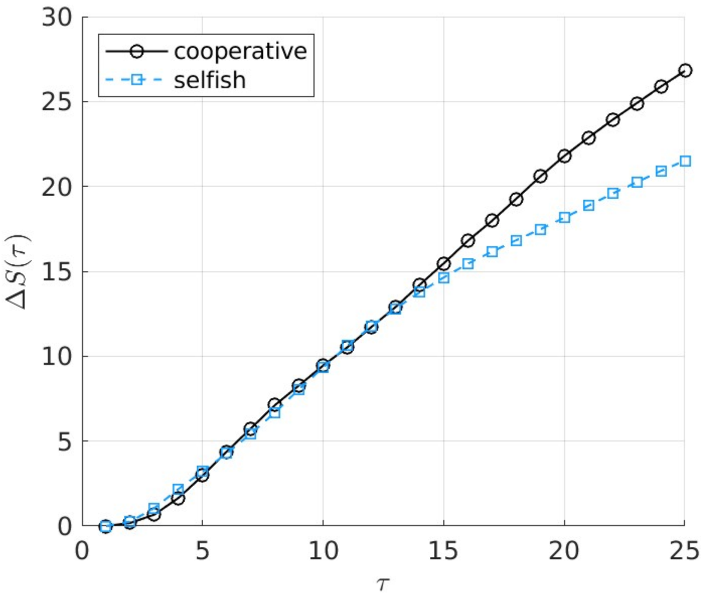


Figure 25. The total scan certainty for **Cooperative** and **Selfish** approaches in Case 2

There exist 2 regions of interest. Region B has high victim density and low scan certainty, making it a priority for exploration. However, extremely low observability makes it challenging for the UAV, while the UGV is better suited despite the high terrain difficulty, which, while significant, is not prohibitive. Region A, on the other hand, offers better observability and lower terrain difficulty than Region B, making it favourable for both agents in terms of ease of movement and scanning effectiveness. In the selfish approach, where there is no global coordination, both the UAV and UGV initially focus on scanning Region A, as it appears favourable to both based on local decision-making. This can be seen in Figure 20. However, as the search progresses, both agents eventually move into the walled region B. This is problematic because the UAV is not well-suited for searching in an area with extremely low observability, making this decision not only energy-inefficient for the UAV but also suboptimal for the overall mission. Alternatively, in the cooperative approach, the supervisory controller assigns regions to each agent to optimise exploration. The UGV is directed toward the enclosed area, or Region B. While the outer Region A has a lower terrain index compared to Region B, making it slightly more favourable in terms of energy consumption for the UGV, the controller prioritises a balanced search strategy to also maximise overall area coverage. Meanwhile, the UAV scans the surrounding Region A, which has better observability and is better suited to its capabilities. Figure 19 shows this behaviour. This allocation ensures that both agents operate in relatively optimal regions.

The impact of this differing search behaviour can be seen in terms of the area coverage, as shown in Figure 28. The cooperative approach manages to achieve a final scan certainty that is 32% higher than the selfish approach.

Further, the victim detection efficiency can be seen in Figure 29. The cooperative approach finds all the victims (i.e, 8 victims) in almost all simulation iterations for the case. The median detection time is 13 time steps. The selfish approach performs worse, with a median of 7 victims detected and taking a median of 14 time steps.

Next, we observe the energy consumption plot in Figure 30. Here, the cooperative approach consumes about 12.8% more energy than the selfish approach. This can be attributed to the fact that the supervisory controller forces the UGV to visit a more sub-optimal region (region B) to maximise victim detection and area coverage.

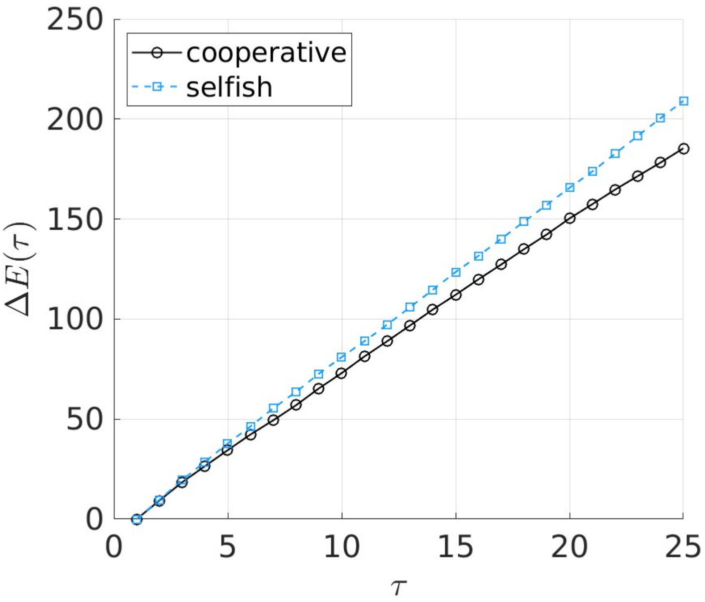


Figure 26. Comparison of Energy Consumption for **Cooperative** and **Selfish** approaches in Case 2

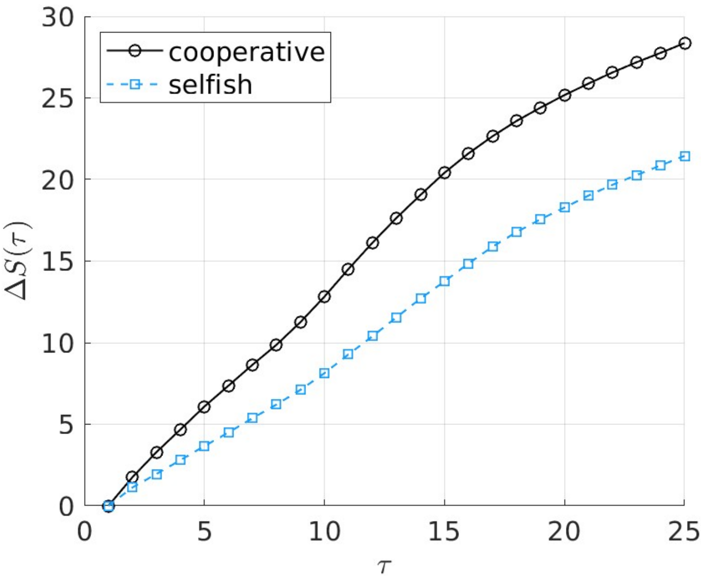


Figure 28. The total scan certainty for **Cooperative** and **Selfish** approaches in Case 3

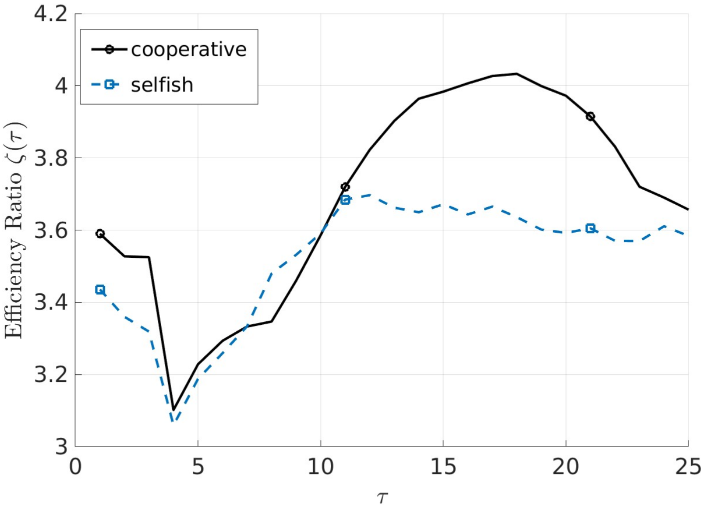


Figure 27. Comparison of Efficiency Ratio for **Cooperative** and **Selfish** approaches in Case 2

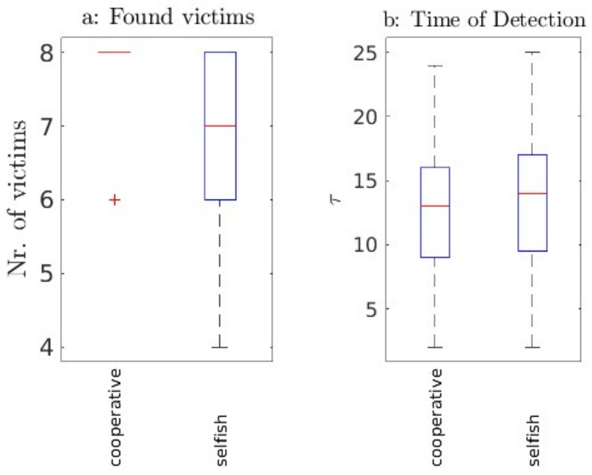


Figure 29. Comparison of Victim Detection Efficiency for **Cooperative** and **Selfish** approaches in Case 3

However, in the selfish case, both agents try to avoid sub-optimal regions to conserve energy for most of the time-steps, but at the cost of the other objectives.

Finally, we compare the Efficiency ratio. Figure 31 suggests that for most time steps (about 60%), the cooperative approach is more efficient than the selfish case, i.e, it balances exploration and energy consumption better.

7. Conclusions

In time-critical disaster scenarios, heterogeneous multi-agent robotic teams integrating the complementary capabilities of aerial and ground vehicles offer significant advantages by enhancing situational awareness and improving navigation within complex indoor environments. This paper presents a hierarchical control framework designed to coordinate multi-agent SaRS. The proposed control strategy effectively integrates the distinct sensory and physical capabilities of individual SaR agents into the mission planning process, aiming to maximise victim detection efficiency, area coverage, and energy utilisation. At the lower level of the control hierarchy, each SaR agent is governed by an autonomous heuristic controller that integrates FLC with a k-shortest path algorithm to independently select targets of interest and corresponding trajectories to those targets. A centralised MPC-based controller at the higher level supervises the overall mission, activating selectively when an agent encounters environmental conditions that render its operational capabilities suboptimal. This hierarchical approach allows the local heuristic controllers to efficiently identify suitable paths and targets under typical conditions. In situations involving agent-environment conflict, the supervisory MPC controller strategically directs agents toward regions optimal for their specific capabilities, thus ensuring global mission effectiveness. Once agents reach suitable operational zones, control is returned to their local heuristic controllers. This method is termed the *cooperative search* approach. Conversely, a strategy relying exclusively on local heuristic control without centralised intervention is referred to as the *selfish search* approach.

In randomised indoor SaR simulations with varying victim distributions, obstacle placements, and environmental conditions (including terrain difficulty and observability), the proposed cooperative mission planning approach consistently demonstrated superior overall performance. Although the ACS method achieved

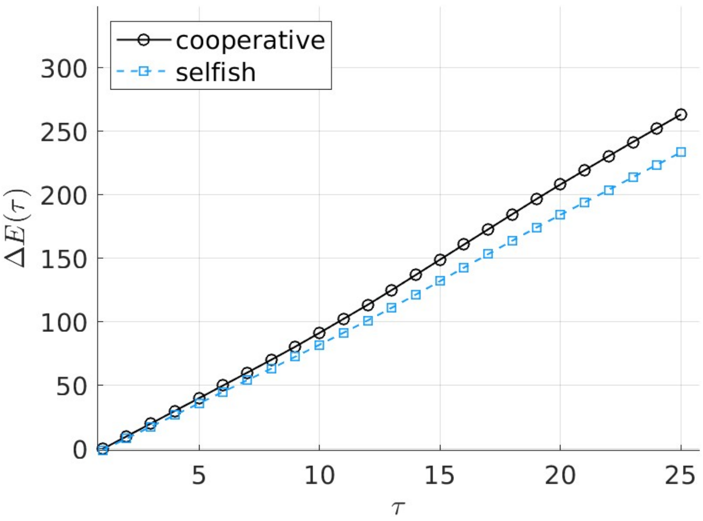


Figure 30. Comparison of Energy Consumption for **Cooperative** and **Selfish** approaches in Case 3

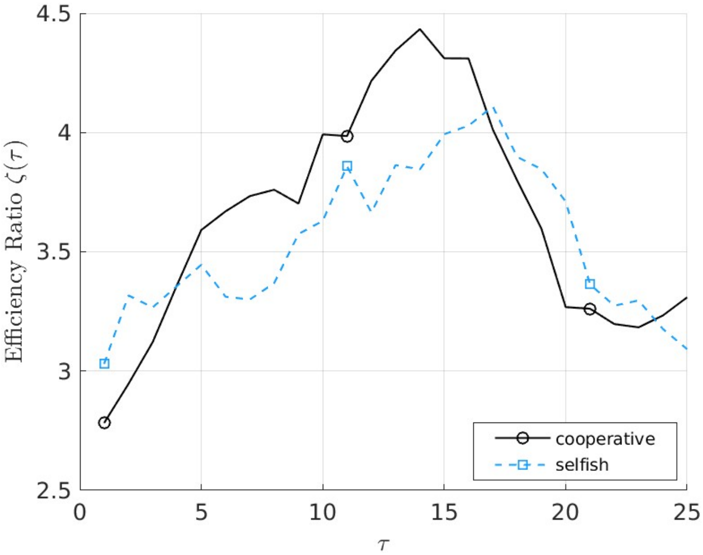


Figure 31. Comparison of Efficiency Ratio for **Cooperative** and **Selfish** approaches in Case 3

the highest scan certainty due to its explicit exploration-oriented design, the cooperative approach provided competitive coverage levels while simultaneously balancing additional critical objectives, such as victim detection and energy efficiency. Specifically, in victim detection tasks, the cooperative strategy exhibited the highest median detection count with the lowest variability, outperforming both the selfish and ACS strategies. Despite requiring slightly more energy compared to the pure-MPC method, the cooperative approach achieved significantly better performance across other critical indicators and maintained a consistently higher efficiency ratio throughout most of the simulation period. Furthermore, simulations conducted under specialised SaR scenarios demonstrated that the cooperative approach resulted in enhanced search behaviour, leading to superior area coverage, victim detection, and overall operational efficiency compared to the selfish approach. These findings highlight that the hierarchical integration of model predictive and supervisory controls, combined with the synergistic leveraging of UAV-UGV capabilities, effectively balances exploration with targeted victim search, offering robust, scalable performance advantages in complex and uncertain indoor SaR environments.

8. Recommendations for Future Research

Future research should aim to incorporate improved models of dynamic environmental elements, including fire propagation, smoke dispersion, explosive hazards, and falling debris. Such enhancements would significantly increase simulation realism, enabling planning algorithms to better address the dynamic and unpredictable conditions frequently encountered in real-world SaR operations. Additionally, addressing the perception component, which is currently assumed rather than explicitly modelled in this research, is critical. Developing and integrating robust perception algorithms capable of accurately sensing and interpreting dynamic indoor environments, including victim detection and hazard identification, will greatly increase the autonomy and applicability of multi-robot SaR systems. Furthermore, developing high-fidelity simulations that closely replicate real-world physics and intricate environmental interactions is essential for validating algorithms under realistic conditions, thus ensuring their reliability prior to deployment. Exploring the scalability of larger multi-robot teams can also provide valuable insights into enhancing cooperative search efficiency. Future work should address associated challenges such as communication constraints, coordination complexity, and computational overhead inherent in large-scale deployments. Additionally, expanding team heterogeneity through the integration of diverse robotic platforms and applying novel task allocation methods could more effectively leverage their distinct capabilities. This would result in enhanced operational flexibility, enabling specialised robots to execute tasks most suited to their strengths. Research can also go into search by larger teams of robots. It is expected that the methodology proposed in this research would scale well with more agents. However, this must be evaluated. Lastly, introducing additional layers of hierarchy or abstraction in control architectures can further optimise the balance between global coordination and local autonomy in complex and uncertain SaR environments.

References

[1] D. P. Stormont and V. H. Allan, “Managing risk in disaster scenarios with autonomous robots”, *Journal of Systemics, Cybernetics and Informatics*, vol. 7, no. 4, pp. 66–71, 2009. [Online]. Available: https://iiisci.org/journal/pdv/sci/pdfs/KR476EN.pdf.

[2] J. Casper and R. R. Murphy, “Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center”, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 33, no. 3, pp. 367–385, Jun. 2003, ISSN: 1083-4419. DOI: 10.1109/TSMCB.2003.811794.

[3] R. D. Arnold, H. Yamaguchi, and T. Tanaka, “Search and rescue with autonomous flying robots through behaviour-based cooperative intelligence”, *Journal of International Humanitarian Action*, vol. 3, no. 1, Dec. 2018. DOI: 10.1186/s41018-018-0045-4.

[4] A. Coburn, R. Spence, and A. Pomonis, “Factors determining human casualty levels in earthquakes: Mortality prediction in building collapse”, 1992, ISBN: 90-5410-060-5. [Online]. Available: https://www.iitk.ac.in/nicee/wcee/article/10_vol10_5989.pdf.

[5] J. Riley and M. Endsley, “The hunt for situation awareness: Human-robot interaction in search and rescue”, *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 48, Sep. 2004. DOI: 10.1177/154193120404800389.

[6] J. P. Queralta, J. Taipalmaa, B. Can Pullinen, *et al.*, “Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision”, *IEEE Access*, vol. 8, pp. 191 617–191 643, 2020. DOI: 10.1109/ACCESS.2020.3030190.

[7] F. Maresca, A. Romero, C. Delgado, V. Sciancalepore, J. Paradells, and X. Costa-Pérez, *React: Multi robot energy-aware orchestrator for indoor search and rescue critical tasks*, 2025. arXiv: 2503.05904 [cs.RO]. [Online]. Available: https://arxiv.org/abs/2503.05904.

[8] M. Lyu, Y. Zhao, C. Huang, and H. Huang, "Unmanned aerial vehicles for search and rescue: A survey", *Remote Sensing*, vol. 15, no. 13, 2023, ISSN: 2072-4292. DOI: 10.3390/rs15133266. [Online]. Available: https://www.mdpi.com/2072-4292/15/13/3266.

[9] T. Ran, L. Yuan, and J. Zhang, "Scene perception based visual navigation of mobile robot in indoor environment", *ISA Transactions*, vol. 109, pp. 389–400, Mar. 2021. DOI: 10.1016/j.isatra.2020.10.023.

[10] D. Paez, J. P. Romero, B. Noriega, G. A. Cardona, and J. M. Calderon, "Distributed particle swarm optimization for multi-robot system in search and rescue operations", *IFAC-PapersOnLine*, vol. 54, no. 4, pp. 1–6, 2021, 4th IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control CESCIT 2021, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2021.10.001.

[11] I. Munasinghe, A. Perera, and R. C. Deo, "A comprehensive review of UAV-UGV collaboration: Advancements and challenges", *Journal of Sensor and Actuator Networks*, vol. 13, no. 6, 2024, ISSN: 2224-2708. DOI: 10.3390/jsan13060081.

[12] S. Y. Ku, G. Nejat, and B. Benhabib, "Wilderness search for lost persons using a multimodal aerial-terrestrial robot team", *Robotics*, vol. 11, no. 3, 2022, ISSN: 2218-6581. DOI: 10.3390/robotics11030064.

[13] K. Asadi, A. Kalkunte Suresh, A. Ender, *et al.*, "An integrated UGV-UAV system for construction site data collection", *Automation in Construction*, vol. 112, p. 103 068, 2020, ISSN: 0926-5805. DOI: https://doi.org/10.1016/j.autcon.2019.103068.

[14] M. Saska, T. Krajnik, and L. Pfeucil, "Cooperative µUAV-UGV autonomous indoor surveillance", in *International Multi-Conference on Systems, Signals and Devices*, 2012, pp. 1–6. DOI: 10.1109/SSD.2012.6198051.

[15] R. Parasuraman, T. B. Sheridan, and C. D. Wickens, "A model for types and levels of human interaction with automation", *IEEE Transactions on systems, man, and cybernetics-Part A: Systems and Humans*, vol. 30, no. 3, pp. 286–297, 2000. DOI: 10.1109/3468.844354.

[16] A. Agha, K. Otsu, B. Morrell, *et al.*, *Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge*, Mar. 2021. DOI: 10.48550/arXiv.2103.11470.

[17] R. Arnold, J. Jablonski, B. Abruzzo, and E. Mezzacappa, "Heterogeneous UAV multi-role swarming behaviors for search and rescue", Aug. 2020, pp. 122–128. DOI: 10.1109/CogSIMA49017.2020.9215994.

[18] Z. Beck, L. Teacy, A. Rogers, and N. R. Jennings, "Online planning for collaborative search and rescue by heterogeneous robot teams", in *Adaptive Agents and Multi-Agent Systems*, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:9955127.

[19] W.-Y. G. Louie and G. N. and, "A victim identification methodology for rescue robots operating in cluttered USAR environments", *Advanced Robotics*, vol. 27, no. 5, pp. 373–384, 2013. DOI: 10.1080/01691864.2013.763743.

[20] R. Almadhoun, T. Taha, L. Seneviratne, and Y. Zweiri, "A survey on multi-robot coverage path planning for model reconstruction and mapping", *SN Applied Sciences*, vol. 1, no. 8, Jul. 2019. DOI: 10.1007/s42452-019-0872-y.

[21] X. Sun, Y. Liu, W. Yao, and N. Qi, "Triple-stage path prediction algorithm for real-time mission planning of multi-UAV", *Electronics Letters*, vol. 51, no. 19, pp. 1490–1492, 2015. DOI: 10.1049/el.2015.1244.

[22] J. Liu, S. Anavatti, M. Garratt, and H. A. Abbass, "Modified continuous ant colony optimisation for multiple unmanned ground vehicle path planning", *Expert Systems with Applications*, vol. 196, p. 116 605, 2022. DOI: https://doi.org/10.1016/j.eswa.2022.116605.

[23] K. Rajchandar, R. Baskaran, P. K. Panchu, and M. Rajmohan, "A novel fuzzy and reverse auction-based algorithm for task allocation with optimal path cost in multi-robot systems", *Concurrency and Computation-Practice and Experience*, vol. 34, no. 5, 2022. DOI: https://doi.org/10.1002/cpe.6716.

[24] H. Bae, G. Kim, J. Kim, D. Qian, and S. Lee, "Multi-robot path planning method using reinforcement learning", *Applied Sciences*, vol. 9, no. 15, 2019, ISSN: 2076-3417. DOI: 10.3390/app9153057.

[25] M. Çetinkaya, *Multi-agent path planning using deep reinforcement learning*, 2021. arXiv: 2110.01460 [cs.LG]. [Online]. Available: https://arxiv.org/abs/2110.01460.

[26] D. Zhu, Y. Liu, and B. Sun, "Task assignment and path planning of a multi-AUV system based on a glasius bio-inspired self-organising map algorithm", *The Journal of Navigation*, vol. 71, no. 2, pp. 482–496, 2018. DOI: 10.1017/S0373463317000728.

[27] M. Popović, J. Ott, J. Rückin, and M. J. Kochenderfer, "Learning-based methods for adaptive informative path planning", *Robotics and Autonomous Systems*, vol. 179, p. 104 727, 2024, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2024.104727.

[28] R. Arnold, J. Jablonski, B. Abruzzo, and E. Mezzacappa, "Heterogeneous UAV multi-role swarming behaviors for search and rescue", in *2020 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*, 2020, pp. 122–128. DOI: 10.1109/CogSIMA49017.2020.9215994.

[29] L. Zhu, J. Cheng, H. Zhang, Z. Cui, W. Zhang, and Y. Liu, *Autonomous and adaptive role selection for multi-robot collaborative area search based on deep reinforcement learning*, 2023. arXiv: 2312.01747 [cs.RO]. [Online]. Available: https://arxiv.org/abs/2312.01747.

[30] K. Jose and D. K. Pratihar, "Task allocation and collision-free path planning of centralized multi-robots system for industrial plant inspection using heuristic methods", *Robotics and Autonomous Systems*, vol. 80, pp. 34–42, 2016, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2016.02.003.

[31] J. Olofsson, C. Veibäck, G. Hendeby, and T. A. Johansen, "Outline of a system for integrated adaptive ice tracking and multi-agent path planning", in *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, IEEE, 2017, pp. 13–18. DOI: 10.1109/RED-UAS.2017.8101636.

[32] M. Lagoudakis, M. Berhault, S. Koenig, P. Keskinocak, and A. Kleywegt, "Simple auctions with performance guarantees for multi-robot task allocation", in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 1, 2004, 698–705 vol.1. DOI: 10.1109/IROS.2004.1389434.

[33] V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, "Ensemble coordination approach in multi-AGV systems applied to industrial warehouses", *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 922–934, 2015. DOI: 10.1109/TASE.2015.2446614.

[34] D. Di Paola, D. Naso, and B. Turchiano, "Consensus-based robust decentralized task assignment for heterogeneous robot networks", in *Proceedings of the 2011 American Control Conference*, IEEE, 2011, pp. 4711–4716. DOI: 10.1109/ACC.2011.5990987.

[35] A. Batinović, J. Oršulić, T. Petrović, and S. Bogdan, "Decentralized strategy for cooperative multi-robot exploration and mapping", *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 9682–9687, 2020, 21st IFAC World Congress, ISSN: 2405-8963. DOI: https://doi.org/10.1016/j.ifacol.2020.12.2618.

[36] Y. Liu and G. Nejat, "Robotic urban search and rescue: A survey from the control perspective", *Journal of Intelligent and Robotic Systems*, vol. 72, no. 2, pp. 147–165, Mar. 2013. DOI: 10.1007/s10846-013-9822-x.

[37] B. B. Werger and M. J. Matarić, "Broadcast of local eligibility for multi-target observation", in *Distributed Autonomous Robotic Systems 4*, Springer, 2000, pp. 347–356, ISBN: 978-4-431-67919-6. DOI: 10.1007/978-4-431-67919-6_33.

[38] C. de Koning and A. Jamshidnejad, "Hierarchical integration of model predictive and fuzzy logic control for combined coverage and target-oriented search-and-rescue via robots with imperfect sensors", *Journal of Intelligent and Robotic Systems*, vol. 107, no. 3, Mar. 2023. DOI: 10.1007/s10846-023-01833-2.

[39] S. D. Kumar, "Hierarchical control for multi-robot systems using contracts", M.S. thesis, Delft University of Technology, Oct. 2024. [Online]. Available: http://resolver.tudelft.nl/uuid:79e2ae59-751b-467a-814d-46c1ce52f61d.

[40] Y. Chen, Y. Li, W. Zheng, and X. Wan, "Transformer fusion-based scale-aware attention network for multispectral victim detection", *Complex and Intelligent Systems*, vol. 10, Jun. 2024. DOI: 10.1007/s40747-024-01515-y.

[41] Y. Yang, M. M. Polycarpou, and A. A. Minai, "Multi-uav cooperative search using an opportunistic learning method", *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 5, pp. 716–728, Jan. 2007, ISSN: 0022-0434. DOI: 10.1115/1.2764515.

[42] A. Elhadary, M. Rabah, E. Ghanim, R. Mohie, and A. Taha, "The influence of flight height and overlap on UAV imagery over featureless surfaces", *MEJ. Mansoura Engineering Journal*, vol. 47, no. 2, pp. 34–42, Oct. 2022. DOI: 10.21608/bfemu.2022.264943.

[43] 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. DOI: 10.1109/TSSC.1968.300136.

[44] E. W. Dijkstra, "A note on two problems in connexion with graphs", *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959. DOI: 10.1007/bf01386390.

[45] J. Y. Yen, "Finding the k shortest loopless paths in a network", *Management Science*, vol. 17, no. 11, pp. 712–716, 1971, ISSN: 00251909, 15265501. [Online]. Available: http://www.jstor.org/stable/2629312.

[46] H. V. Abeywickrama, B. A. Jayawickrama, Y. He, and E. Dutkiewicz, "Empirical power consumption model for UAVs", in *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, 2018, pp. 1–5. DOI: 10.1109/VTCFall.2018.8690666.

[47] A. S. El-Kabbany and A. Ramirez-Serrano, "Terrain roughness assessment for human assisted ugv navigation within heterogeneous terrains", in *2009 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2009, pp. 1501–1506. DOI: 10.1109/ROBIO.2009.5420936.

[48] A. E. Carvalho, J. F. Ferreira, and D. Portugal, "3D traversability analysis and path planning based on mechanical effort for UGVs in forest environments", *Robotics and Autonomous Systems*, vol. 171, p. 104 560, 2024, ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2023.104560.

[49] N. Fazal, M. T. Khan, S. Anwar, J. Iqbal, and S. Khan, "Task allocation in multi-robot system using resource sharing with dynamic threshold approach", *PLOS ONE*, vol. 17, no. 5, e0267982, May 2022. DOI: 10.1371/journal.pone.0267982.

Part II

Literature Review

2
Introduction

Human lives and societies are profoundly impacted by disastrous natural events like earthquakes, floods, and hurricanes, as well as human-induced incidents such as industrial accidents and urban fires. From 2000 to 2019, a total of 7,348 major disaster events were recorded globally, resulting in approximately 1.23 million deaths, impacting 4.2 billion individuals, and causing an estimated $2.97 trillion in economic losses. On average, disasters claim between 40,000 to 50,000 lives annually [1]. Beyond the immediate devastation, these events contribute to long-term consequences such as population displacement, public health emergencies, and sustained economic disruption. According to the UNDRR a disaster is defined as [2]:

> A serious disruption of the functioning of a community or a society at any scale due to hazardous events interacting with conditions of exposure, vulnerability and capacity, leading to one or more of the following: human, material, economic and environmental losses and impacts.

Khorram-Manesh et al. offer an extensive analysis of disaster management, consistent with the framework outlined by the United Nations Office for Disaster Risk Reduction (UNDRR) [3]. They underscore that although events such as floods, wildfires, and pandemics are frequently unavoidable, their impact can be significantly reduced through preparedness and effective response mechanisms. This underscores the importance of well-coordinated response and recovery strategies to mitigate the aftermath of such events. Central to this is the provision of victim assistance and relief, which necessitates the rapid deployment of specialised rescue teams to affected regions.

3

Search and Rescue Robotics

Search and Rescue (SaR) scenarios can broadly be categorised into outdoor and indoor (or urban) disaster environments. Outdoor SaR typically involves wide-area search operations in open or semi-structured settings such as forests, mountains, or regions impacted by disasters. In contrast, Indoor SaR involves navigating confined or structurally compromised spaces, such as collapsed buildings, underground tunnels, mines, or industrial facilities. Historically, SaR missions have relied heavily on human responders and trained rescue dogs, who, despite their effectiveness, encounter considerable limitations when facing hazardous environments or expansive search areas [4]. Human teams often face risks from dangerous conditions like toxic environments and unstable debris, alongside limitations due to physical fatigue and restricted access. Rescue animals, although highly capable of locating victims, can only operate for limited periods and require direct handler supervision. Such constraints underscore the need for robotic systems, which can mitigate human risks and extend the reach and duration of SaR operations.

The use of robotic platforms in SaR has rapidly expanded over recent years. For example, following the 2010 Haiti earthquake, ground robots were effectively utilised within collapsed structures to locate survivors [5]. Unmanned Aerial Vehicles (UAVs) have increasingly been deployed for tasks such as mapping disaster sites, locating survivors, and delivering supplies, offering significant advantages in speed and coverage compared to ground-based teams [6]. A notable instance includes UAVs being employed to locate missing individuals in the Himalayan region, effectively surveying vast terrains at high altitudes [7].

Indoor or urban SaR operations present particularly challenging environments for robotic systems. These settings typically lack GPS signals, are dark, confined, and extensively unstructured due to structural damage. Robots operating indoors must navigate obstacles like debris piles, narrow passages, and unstable floors, with a substantial risk of entrapment. Communication technologies, including GPS and radio signals, are typically weak or unavailable deep within collapsed structures or underground spaces. Thus, indoor SaR robots rely primarily on onboard systems for navigation and localisation. Additionally, indoor scenarios often include multi-level structures and unstable elements . Small UAVs, such as quadrotors, have shown potential in these environments, providing access to spaces unreachable by ground robots, though they face significant challenges, including GPS-denied navigation, collision risks in confined areas, and disturbances from airflow interactions with structural elements [8]. Conversely, Unmanned Ground Vehicles (UGVs) are particularly beneficial in indoor scenarios, where they can safely explore deeper into hazardous locations, such as collapsed basements or industrial accident sites, inaccessible to human rescuers.

Recently, multi-robot systems that integrate different robotic platforms have emerged as a powerful strategy in SaR operations. The use of multiple robots enhances the coverage, speed, and robustness of SaR missions, enabling more thorough and efficient search efforts [9]. These multi-robot systems can either be homogeneous or heterogeneous. In homogeneous systems, robots share similar capabilities, sensors, and performance attributes. Conversely, heterogeneous systems consist of robots with diverse functionalities, allowing specialisation and improved efficiency for various tasks within a SaR operation. UAVs and UGVs, when collaboratively deployed, can significantly outperform single-platform approaches. For instance, UAVs can quickly perform aerial reconnaissance, map terrain and identify obstacles, providing vital information that helps UGVs optimise their ground navigation and search paths. Santamaria-Navarro et al. demonstrated such a cooperative exploration system, where UAVs provided real-time aerial perspectives, enhancing the situational awareness and operational efficiency of UGVs navigating on the ground [10]. Additionally, UAV-UGV collaborations can yield comprehensive multi-scale mapping. UAVs excel at rapidly generating broad-scale maps of areas, albeit at lower resolution, while UGVs capture detailed local maps with ground-level sensors, such as LiDAR. Recent research has shown that aligning aerial maps from UAVs with ground maps from UGVs can significantly enhance traversability mapping, improving ground robot navigation [11]. Field trials, such as the deployment described by Kruijff-Korbayová et al., where ground and aerial robots collaborated in assessing earthquake-damaged buildings in Italy, confirm the practical benefits and enhanced operational capabilities of combined UAV-UGV systems [12].

Effective multi-robot coordination involves complex mission planning, which can be decomposed into two core problems: task allocation and trajectory planning. Task allocation is particularly crucial in scenarios with more tasks than available robots, requiring prioritisation based on the relative capabilities of each robot. The heterogeneity of robotic systems adds complexity to task allocation, as different robots may be better suited for distinct tasks. The second aspect, trajectory planning, involves determining optimal routes for robots to reach their designated tasks, which is particularly challenging in dynamic, unpredictable environments like disaster zones, where conditions constantly evolve. The need for continuous real-time planning to adapt to environmental changes highlights the necessity for advanced mission planning methods capable of addressing the dynamics and uncertainties inherent in SaR scenarios.

3.1. Project Scope

The literature survey presented in this thesis supports research aimed at developing a mission planning control approach designed to reduce the time required to locate victims and enhance environmental understanding in urban, indoor disaster scenarios. The central focus of this thesis is the design of an adaptive control strategy specifically tailored for a heterogeneous collaborative system comprising UAVs and UGVs. Given the inherent uncertainty in disaster environments, such as unpredictable victim dispersion and hazardous conditions like structural damage, fires, and smoke, this problem is particularly challenging. The proposed approach will aim to leverage the distinct capabilities of each robotic agent to allocate search tasks efficiently and adaptively, using real-time data gathered from the environment during operations. The mission planning controller will be designed to progressively learn about its environment and optimise its search strategy as new environmental information is obtained. The thesis is structured into two primary stages: firstly, the development of the mission planning control approach, followed by its implementation and evaluation within a realistic simulation environment. This simulation will demonstrate and assess the effectiveness and adaptability of the developed mission planning module in realistic SaR operations.

4

Control Approaches

Prior to conducting a comparative review of various SaRS, it is essential to establish an understanding of the common control approaches. This chapter presents a review of selected heuristic and model-based control approaches, outlining their fundamental principles, characteristics, and practical applications.

4.1. Heuristic Methods

In the context of search and optimisation problems, a heuristic method employs rules of thumb, educated guesses, or approximations to guide the search process toward a satisfactory solution. These heuristics are often based on knowledge about the specific problem domain, such as typical layouts of indoor environments, general problem-solving strategies that have been observed to be effective in similar situations or simplified models/approximations of the problem. These search methods do not guarantee finding the absolute best solution but aim to find good solutions within a reasonable time frame. In large and cluttered environments, exploring every possible path is computationally infeasible. Heuristics help narrow down the search space, by focusing only on promising paths. In this study we mainly evaluate two classes of heuristic approaches: (1) Learning-based and bio-inspired methods including reinforcement learning and swarm intelligence algorithms (Ant Colony, PSO, GA) that adapt or evolve solutions for multi-robot coordination, and (2) Fuzzy logic controllers (FLC), which encode human-like rules for decision-making under uncertainty. We also study a few other approaches like A* and Djikstra's algorithm.

4.1.1. Fuzzy Logic

The earliest work in terms of establishing a formal mathematical framework for Fuzzy logic can be attributed to Zadeh [13]. Furthermore, Dubois offers a comprehensive and in-depth examination of fuzzy logic and fuzzy systems in [14]

Fuzzy logic controllers use fuzzy set theory to manage uncertainty and imprecision inherent in sensor data. In classical set theory, sets are defined in binary terms, i.e. an element x either belongs to a set A or does not, with no intermediate degrees of membership. Mathematically, this translates to a membership function of the form:

$$\mu_A = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \tag{4.1}$$

For vaguely defined concepts such as the 'set of damaged buildings' or the 'set of fast cars', it is often unclear which elements definitively belong to the set. This ambiguity arises because such sets are described using linguistic, human-centric terms rather than precise, quantitative definitions. Despite their imprecision, these types of concepts are fundamental to the way humans reason, take decisions and act. To mathematically represent this inherent uncertainty, fuzzy set theory was introduced. Unlike classical sets, fuzzy sets allow elements to possess varying degrees of membership, as can be seen below:

$$\mu_A = \begin{cases} 1 & \text{x is a member of A with complete confidence} \\ 0 < x < 1 & \text{x is a member of A with partial confidence} \\ 0 & \text{x is not a member of A with full confidence} \end{cases} \tag{4.2}$$

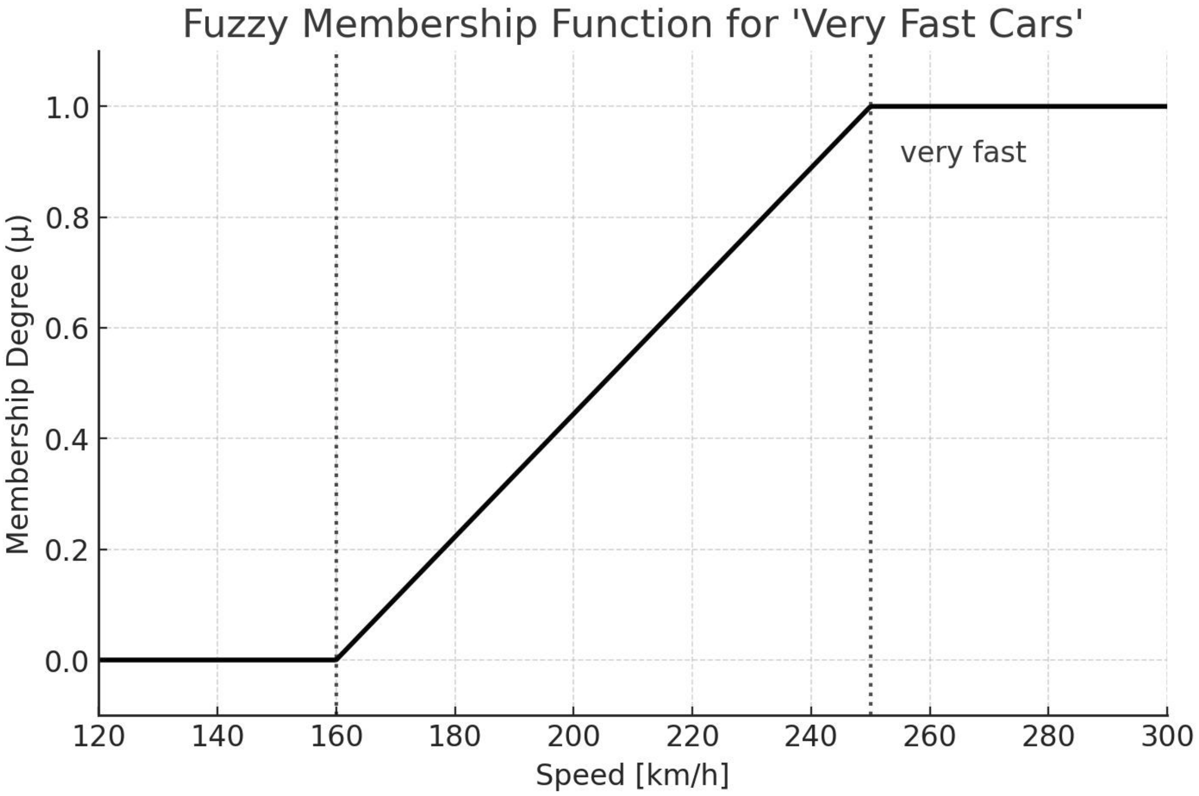


Figure 4.1: Membership Function for set of 'very fast cars'

We can consider the "set of fast cars" as an example. The term 'fast' is inherently subjective and context-dependent, reflecting personal perceptions rather than precise measurements. For instance, someone who regularly drives high-performance sports cars might have a very different perception of what qualifies as fast compared to someone who drives a compact city car. To model this variability, a membership function can be defined to assign degrees of membership based on perceived speed. For example, one might argue that any car with a top speed below 160 km/h is definitely not fast, while anything exceeding 250 km/h definitely is. The speeds in between (160 km/h to 250 km/h) represent a fuzzy zone, where cars have partial membership to the "fast-cars-set", depending on individual interpretation. The membership function for this could look like Figure (4.1).

Thus, instead of crisp thresholds, FLCs use continuous membership functions to represent vague concepts like "near an obstacle" or "high battery level" with degrees of truth between 0 and 1. The controller encodes expert knowledge as if-then rules (e.g., "IF obstacle is very close in front AND speed is high THEN turn sharply"). These rules are evaluated using a fuzzy inference engine and then defuzzified to produce a concrete control output (such as a steering angle or throttle command) [15]. The controller comprises of the following components:

1. **Rule base:** A set of fuzzy IF-THEN rules capturing the expert knowledge on how to control the system. For example, a rule might state IF distance_to_obstacle is Small AND heading_error is Small THEN turn Slightly Left.
2. **Inference Mechanism:** The logic that evaluates which rules are triggered by the current fuzzy inputs and how to aggregate their recommendation. Often, Mamdani or Takagi-Sugeno inference is used to combine rule effects.
3. **Fuzzification:** A pre-processing step that converts crisp sensor readings into fuzzy values (membership grades in fuzzy sets) so they can be used in the rule conditions. For instance, a distance of 1.5 m might correspond to a fuzzy value of 0.8, which in turn corresponds to "Small".
4. **Defuzzification:** A post-processing step that converts the fuzzy control conclusions back into a crisp output command (e.g., control actions suggested by rules).

As mentioned earlier, Mamdani and Takagi-Sueno are commonly used inference methods and will be discussed below.

Mamdani Inference
It is a type of *linguistic fuzzy model*, which has a rule base of the form:

$$R_i : \text{if x is } A_i \text{ then } y_i \text{ is } B_i \quad i = 1, \ldots, K$$

This system was introduced by Ebrahim Mamdani in 1975 [16]. Here both input and output variables are represented by fuzzy sets. The algorithm first begins with converting crisp input values into degrees of membership across relevant fuzzy sets using membership functions. For an input variable, x and a fuzzy set A, the membership function $\mu_A(x)$ determines the degree to which x belongs to A. This fuzzification is done in the following manner:

$$A' = \left\{ \frac{\mu_A(x)}{x} \right\} \quad \text{where x belongs to set of inputs}$$

Next, the degree of fulfilment or firing strength β or each rule is calculated by evaluating the maximum value of the intersection between the input membership function $\mu'_A(x)$ and the membership function of each rule in the rule base $\mu_{A_i}(x)$.

$$\beta_i = \text{max } [\mu'_A(x) \cap \mu_{A_i}(x)] \quad \forall i \in [1, K]$$

Following this step, individual fuzzy output sets are generated by computing the intersection of the corresponding firing strength with the output membership function associated with that rule.

$$B_i : \mu'_{B_i}(y) = \beta_i \cap \mu_{B_i} \quad \forall i \in [1, K]$$

This is followed by aggregation of all individual output sets into a single fuzzy set B':

$$B' : \mu'_B(y) = \text{max } \mu_{B'_i}(y) \quad \forall i \in [1, K]$$

Finally, the aggregated fuzzy output is converted into a crisp value. A common method to do so is the centroid (center of gravity) approach:

$$y'_{crisp} = \frac{\sum_{i=1}^{K} \mu_{B'}(y_i) y_i}{\sum_{i=1}^{K} \mu_{B'}(y_i)}$$

Takagi-Sugeno Inference
It is an inference system whose rule structure is represented as follows:

$$R_i : \text{if x is } A_i \text{ then } y_i \text{ is } f_i(x) \quad \forall i = 1, \ldots, K$$

This system was developed by Takagi and Sugeno in 1985 [17]. This approach is distinct from the Mamdani method in the sense that here the output is a function of the input variables, rather than fuzzy propositions. Similar to the Mamdani approach, the crisp inputs are fuzzified and the degree of fulfilment or firing strength β_i is computed for all the rules. The output function values are computed and their weighted average (with the degree of fulfilment value β_i as weights), gives the crisp output values:

$$y'_{crisp} = \frac{\sum_{i=1}^{K} \beta_i f(x_i)}{\sum_{i=1}^{K} \beta_i}$$

Overall, in the Mamdani FIS outputs are fuzzy sets requiring defuzzification to obtain crisp values. In the Takagi-Sugeno FIS outputs are typically linear or constant functions of input variables, resulting in crisp outputs without the need for defuzzification. This makes it computationally more efficient. However, Mamdani FIS is more intuitive and better suited for systems requiring human interpretability and expert knowledge integration. On the other hand, TS is better suited for mathematical analysis and optimisation.

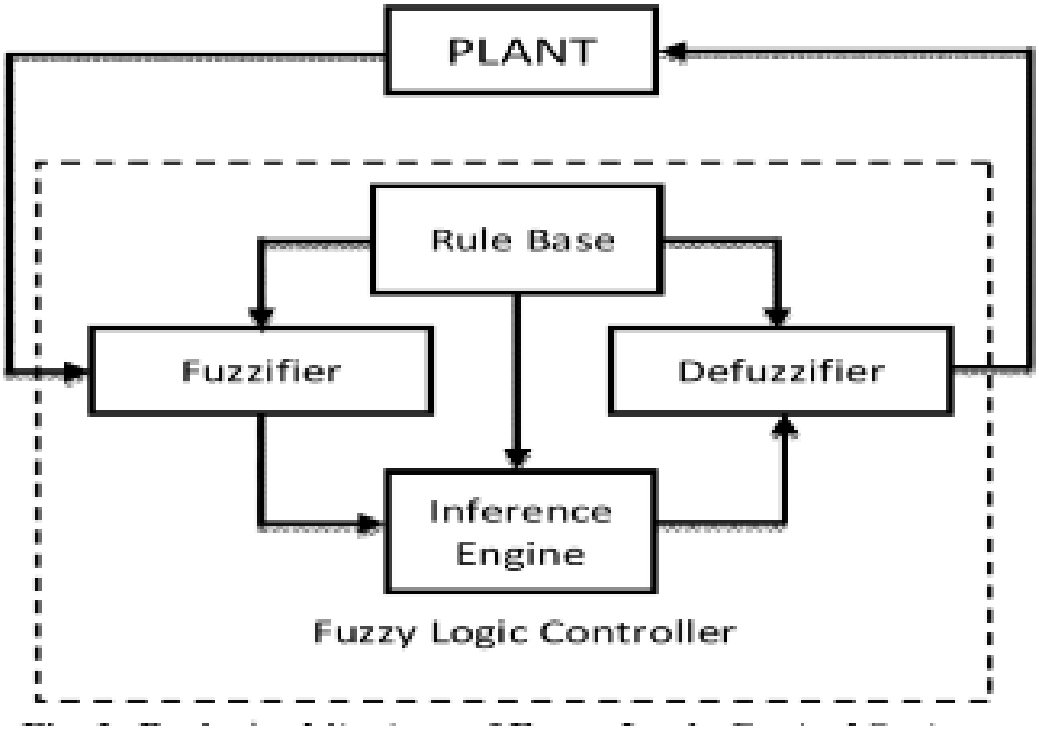


Figure 4.2: Direct FLC Architecture [19]

4.1.2. Fuzzy Logic Controller

As mentioned previously, FLC controllers have gained prominence in control systems for their ability to handle uncertainty and incorporate heuristic expert knowledge. Unlike crisp logic, fuzzy logic allows partial truth values, enabling smooth control actions even with imprecise sensor data or nonlinear system dynamics [18]. In robotics, particularly in SaR scenarios, environments are often unstructured and information is noisy or incomplete. FLCs are well-suited to such conditions, providing robust decision-making where traditional linear controllers struggle. In addition to its capability for handling non-linear control problems, FLC offers a highly interpretable rule structure. This transparency is beneficial during design and tuning, as it allows the developer to clearly trace and understand the sources of knowledge and decision logic embedded within the rule base. In contrast, black-box methods like artificial neural networks (ANNs) often lack this clarity, making it more difficult to interpret the internal decision-making processes. In this report, we examine three categories of fuzzy control strategies: **Direct Fuzzy Logic Control (DFLC)**, **Supervisory Fuzzy Logic Control (SFLC)**, and **Model-Based Fuzzy Logic Control (MBFLC).**
First, we consider Direct Fuzzy Logic Controllers. Here, the controller directly maps sensor inputs to control outputs via linguistic rules. The block diagram in Figure (4.2) shows this control architecture.

This structure involves fuzzification or converting crisp inputs (e.g. distances, angles, velocities) into fuzzy variables using membership functions. This is followed by applying an FIS (often Mamdani inference) to determine the fuzzy control action (based on set of defined if-then rules). Finally, the inferred fuzzy output is converted back to a crisp actuator command (e.g. wheel speed or steering angle). Such DFLCs have been applied in numerous works including swarm control in an unknown SaR environment [20], and process control in industries [21]. Next, we consider Supervisory Fuzzy Logic Controllers (SFLCs), which involves a two-layer control architecture: a high-level fuzzy supervisor adjusts or coordinates one or more low-level controllers. The typical structure, as shown in Figure (4.3) of SFLC consists of a primary controller (or multiple controllers) handling the basic control loops and a fuzzy supervisory layer monitoring system performance and context. The supervisor takes as inputs high-level variables (e.g. tracking error, oscillation level, environment state, inter-robot distances) and outputs adjustments like “increase PID gains,” “switch leader robot,” or “trigger emergency stop.” It is important to note that the structure of the fuzzy controller itself remains the same. However, it no longer directly gives the control command. A common application is fuzzy gain scheduling or tuning of conventional controller parameters [22]. In the domain of multi-robot SaR missions, one implementation used a central fuzzy supervisor to allocate search areas to different robots by evaluating each robot's capability and workload [23].

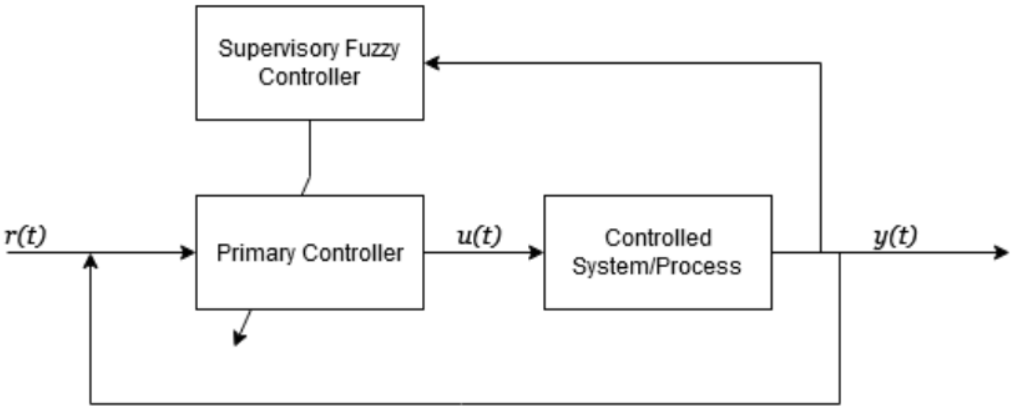


Figure 4.3: Supervisory FLC Architecture

Finally, we studied Model-Based Fuzzy Logic Controllers. MBFLCs integrates fuzzy logic systems with mathematical models of the plant or robot being controlled. Unlike traditional fuzzy controllers that rely purely on rule-based heuristics, MBFLC enhances control performance by incorporating system dynamics—typically through Takagi–Sugeno (T–S) fuzzy models or predictive modelling techniques. These models represent nonlinear systems as interpolations of multiple linear sub-models, enabling the controller to apply a tailored response based on the current operating region. A typical MBFLC system operates by determining the degree to which current system conditions match each local model (via fuzzy membership functions), and then blending corresponding control laws accordingly. This method enables smooth, adaptive control across varying conditions [24].

4.1.3. Learning-based and Adaptive Methods

Researchers have explored learning-based controllers that can improve over time or adapt to the environment. Two major categories here are: Reinforcement Learning (RL) (including deep reinforcement learning, DRL) and bio-inspired optimisation algorithms (such as Ant Colony Systems, Particle Swarm Optimisation, and Genetic Algorithms). These methods often serve as heuristics when an exact solution is unattainable, and many can be integrated into hierarchical frameworks.

RL allows robots to learn decision-making policies by interacting with the environment and receiving feedback in the form of rewards. In an SaR context, an RL agent could receive a positive reward for finding a victim or covering a new area, and a negative reward for time elapsed or getting stuck. Over many trials (simulations or real runs), the agent tunes its policy to maximise the expected cumulative reward. This framework is appealing for SaR because it does not require an explicit model of the environment – the robot learns what to do from experience. Modern DRL uses deep neural networks as function approximators, which enable learning of complex policies from high-dimensional inputs (e.g., LiDAR scans, camera images). One early example applied to multi-robot SaR is a hierarchical reinforcement learning (HRL) architecture proposed by Liu et al. [25]. In their approach, a centralised higher-level agent learned to allocate tasks to each robot (e.g., "Robot A search room 1, Robot B go to room 2") while lower-level controllers executed those tasks. The HRL framework allowed the multi-robot team to collectively decide which rescue tasks to do at a given time and which robot should do them, with the objective of optimising overall exploration efficiency and victim identification. Notably, this learning-based approach led to cooperative behaviour: the robots learned policies for task allocation that improved coverage and avoided duplication of effort. In recent years, deep RL has been applied to SAR problems with increasing success. For instance, Niroui et al. used a deep Q-network to train a rescue robot for autonomous exploration in unknown, cluttered environments [26]. The robot's LiDAR and pose were inputs to a neural network that output motion commands, with the goal of maximising area explored under time constraints. The trained policy learned intelligent exploration strategies (like following walls and systematically turning at intersections), similar to human strategies, but emerged from the RL process. Another example is the use of a decentralised multi-agent RL framework for coordinated exploration [27]. In their setup, each robot runs an RL policy that decides its next move based on its local map (as an occupancy grid) and additional features like frontier distances. Training RL controllers, especially in multi-robot scenarios, is computationally intensive and data-hungry. The state-space grows exponentially with the number of robots (each robot's pose and sensor readings), making convergence difficult. Techniques like centralised training with decentralised execution (CTDE) are often used, where a centralised trainer has access to all robots' states during training to ease learning, but the execution policy is distributed across robots [28]. Also, safety is a major concern – an RL agent might need to explore potentially dangerous actions to learn, which is unacceptable in physical SaR robots. Parallely, there is sufficient interest in applying bio-inspired algorithms to coordinate multi-robot teams. These methods are typically optimisation techniques that iteratively improve a population of candidate solutions, loosely inspired by natural processes like ant foraging, bird flocking, or evolution. They do not "learn" a policy per se (as in RL), but rather search for good solutions. PSO is inspired by the social behaviour of birds flocking. It searches for the optimal path by iteratively improving candidate solutions (particles) based on their own experience and the experience of the swarm. For instance, each robot could be a particle proposing a path or next waypoint, and they iteratively adjust plans to improve a global objective like the total area covered [9]. However, a common challenge with PSO is that it is effective for global optimisation but may converge prematurely to local optima. Further, GAs optimise by simulating evolution where candidate solutions (or chromosomes) undergo selection, crossover, and mutation to produce new offspring solutions, iteratively improving the population. In multi-robot missions, a chromosome might encode a set of routes for all robots or an assignment of tasks to robots. GAs have been used for problems like multi-robot task allocation and path planning, which are often combinatorial. For example, in one research, the multi-robot routing problem was formulated as a variation of the multiple travelling salesman problem and applied GA to evolve good route assignments [29]. Another work by Hayat et al. used a GA to coordinate multiple UAVs in a large-scale outdoor SaR scenario [30]. They encoded different mission tasks and evolved assignments so that the team of drones could cover multiple objectives efficiently. While GAs can handle complex search spaces, they can be computationally expensive [31]. ACO, on the other hand, is inspired by how real ants find shortest paths using pheromone trails. In a robotics context, virtual pheromones can be laid on a grid or graph representation of the search area. Multiple agents (artificial ants) construct paths from a start state to a goal state based on pheromone intensity (probability) and heuristic preferences (like favouring unexplored areas). Through many iterations, pheromone updates guide the algorithm toward high-quality paths or search schedules. In SaR, ACO has been used for multi-robot path planning and area coverage. It is robust and can handle dynamic environments but can be slow to converge, especially for large search spaces. The above heuristic methods are often applied to NP-hard problems. For instance, optimal multi-robot search or routing can be NP-hard, which justifies using metaheuristics like ACO, PSO, or GA to find good feasible solutions in reasonable timeframes. These algorithms do not guarantee a global optimum and have tuning parameters (e.g., number of ants/particles, number of generations) that affect computational load. Another challenge is scaling to many robots or large areas as heuristics may struggle as state space grows.

One of the most commonly used graph-based heuristic search methods is Dijkstra's Algorithm. It works by iteratively exploring nodes starting from the source, always selecting the node with the lowest cumulative cost. While DA is simple and effective for smaller problems, it becomes computationally expensive for larger graphs. A* Algorithm is an informed search algorithm that improves upon DA by incorporating a heuristic function to estimate the cost from the current node to the goal. This guides the search towards promising nodes, enhancing efficiency. While A* is generally more efficient than DA, its performance is highly dependent on the accuracy of the heuristic function [32].

4.2. Optimisation-based Methods

Optimisation-based control treats decision-making as a mathematical optimisation problem, explicitly maximising or minimising objective functions subject to constraints. They rely on models of robot dynamics and the environment, using solvers to compute control actions. One such approach is Model Predictive Control or MPC.

4.2.1. Model Predictive Control

MPC is an advanced control technique that solves a constrained optimal control problem on a rolling horizon. Figure (4.4) shows the architecture for a simple MPC.

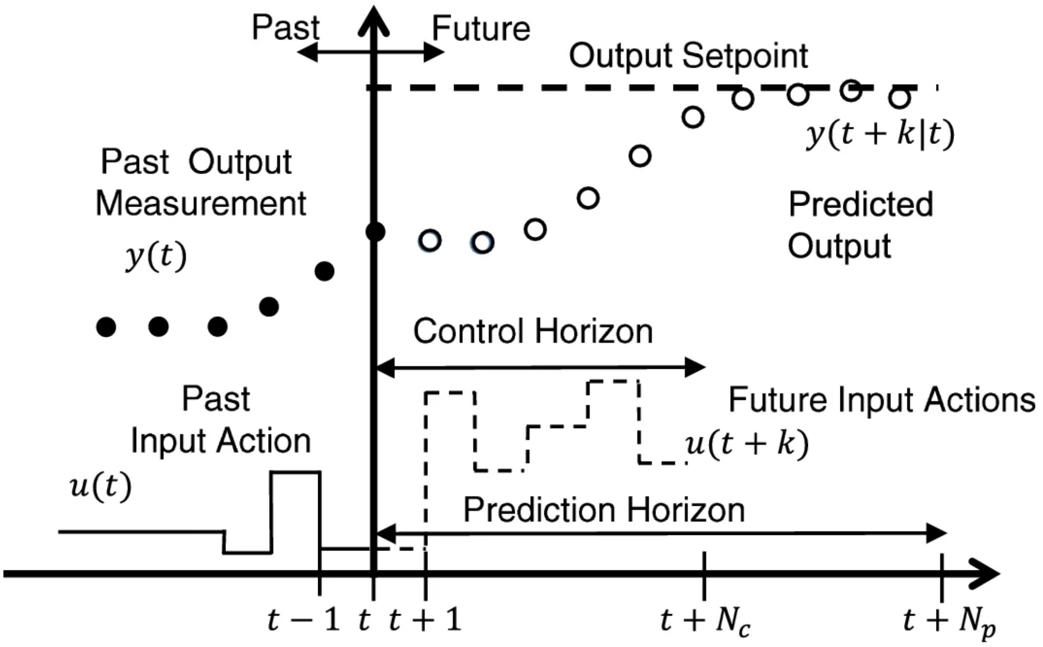


Figure 4.4: MPC architecture [33]

At each time step, an MPC planner optimises a sequence of control inputs over a future horizon (e.g. next N seconds) to minimise a cost function (e.g. time to find victims, deviation from desired path) while respecting system dynamics and constraints. Only the first control input from this optimised sequence is executed, and the optimisation is repeated at the next time step using updated state information. Mathematically, for a discrete-time linear system represented by:

$$x_{k+1} = A_{x_k} + B_{u_k}$$

and having the output equation as:

$$y_k = C_{x_k} + D_{u_k},$$

The MPC controller tries to minimise an objective function of the form:

$$J = \sum_{j=1}^{N_p} \delta_j [\hat{y}_{k+j} - w_{k+j}]^2 + \sum_{j=1}^{N_c} \lambda_j [\triangle u_{k+j-1}]^2$$

subject to:

$$x_{k+1} = f(x_k, u_k)$$
$$y_k = h(x_k, u_k)$$
$$\pi_k \in \mathcal{U}$$

Here, the goal is that the future process output tracks a reference signal w along the horizon while penalising the control effort Δu for doing so. Both δ_j and λ_j are weights for the error and control effort along the horizon. N_p and N_c represent the prediction and control horizon, respectively [33]. These constraints ensure that both the state and output vectors comply with the inherent dynamics of the system, and that the control sequence, π lies within the admissible control space. Essentially, MPC reformulates the control task into a finite-horizon optimisation problem, which can be addressed using various techniques. One such approach is dynamic programming (DP). It solves complex problems by breaking them down into simpler subproblems, solving each of them once, and then storing their solutions. It is ideally suited for sequential decision-making processes. However, as the number of dimensions or state variables increases, the state space grows exponentially, making DP computationally infeasible for high-dimensional problems. Derivative-Free Optimisation methods like pattern-search, find use in problems where gradient information is unavailable or impractical to obtain. DFO methods have been applied in MPC to handle systems with non-differentiable dynamics or cost functions [34][35]. However, they require a substantial number of function evaluations to explore the search space adequately and, hence, can sometimes be computationally expensive. In fact, MPC in itself is a computationally demanding approach. This limitation becomes prohibitive for large-scale systems. Thus, adapted versions like distributed and decentralised control architectures have been explored.

Decentralised MPC operates without strict communication between individual subsystems, making it suitable for decoupled or loosely coupled systems. In this approach, each local controller optimises its own variables independently, without considering the control inputs of other controllers. This leads to simpler implementation but may reduce system performance if the subsystems are more tightly coupled than assumed [36]. In contrast, **Distributed MPC** involves some level of information exchange between local controllers, allowing each to be aware of others' behaviours. This structure is beneficial for strongly coupled systems, where optimisation by one controller influences others. Systems can be fully connected, with all controllers communicating with each other, or partially connected, where communication occurs only among certain subsets. Additionally, distributed MPC can be categorised based on the optimisation behaviour of local controllers. In a **Cooperative Control** framework, multiple local controllers work together to optimise a common, global cost function. Alternatively, if the local controllers optimise their own cost functions, the approach can be said to be **Non-cooperative** [37].

4.3. Hierarchical Controllers

Integrating multiple control strategies can significantly improve system performance, especially in scenarios involving complex, nonlinear, or uncertain dynamics. This section explores several hierarchical controllers that merge various previously discussed control approaches.

4.3.1. Fuzzy Logic Controller with Model Predictive Control

Fuzzy Model Predictive Control (FMPC) integrates fuzzy logic with MPC to manage complex, nonlinear systems effectively. This hybrid approach leverages the strengths of both methodologies: the capacity of fuzzy logic to handle uncertainties and approximate reasoning and the predictive optimisation of MPC. Here, the system's nonlinear dynamics are represented using fuzzy modelling techniques, particularly the Takagi–Sugeno (T-S) fuzzy model. This model approximates a nonlinear system by blending multiple linear models, each of which is associated with specific operating conditions defined by fuzzy sets. The overall system behaviour is obtained by aggregating the contributions from all rules, weighted by their respective membership functions [38]. MPC utilises this fuzzy model to predict future system behaviour over a finite horizon. Another interesting application of FMPC was seen in [39], where it is applied in the context of multi-UAV cooperative target search. In their framework, each UAV maintains and updates a probability map of the target's location, which serves as a fuzzy representation of environmental uncertainty. This map is used to compute a ”future gain” term—representing expected information value—which is added to the MPC cost function. The MPC then optimises this cost function and plans a suitable control strategy. Thus, here instead of determining the system model/behaviour, the fuzzy model modifies the cost function of the MPC.

4.3.2. Fuzzy Model Reference Learning Controller

Fuzzy Model Reference Learning Control (FMRLC) is an adaptive control architecture that combines the intuitive decision-making capability of fuzzy logic with the self-tuning capabilities of Model Reference Adaptive Control (MRAC). This approach was first introduced by Layne and Passino [40]. FMRLC utilises FLC as the primary control component, in combination with a learning mechanism, as shown in Figure (4.5). A reference model generates the desired output trajectory, y_m, based on pre-defined performance objectives. The actual system output, y is then compared against this reference output, and any deviation is interpreted as a control error $e(kt)$ that must be corrected. The fuzzy controller receives this error and its derivative as input. The objective if the FLC is to reduce this error. For this it possesses a learning mechanism composed of a fuzzy inverse model and a knowledge-base modifier. The fuzzy inverse model

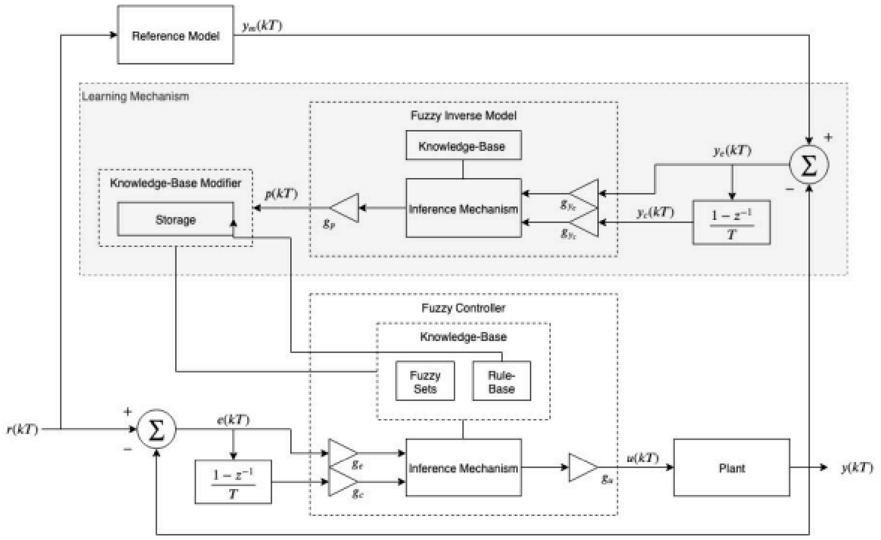


Figure 4.5: FMRLC architecture [40]

estimates the necessary adjustments in control input needed to reduce the output error to zero, while the knowledge-base modifier translates these adjustments into updates to the rule base or output membership functions of the FLC. Over time, this process allows the controller to refine its behaviour iteratively and autonomously to effectively adapt to changes in system dynamics. This learning ability addresses one of the main limitations of conventional FLCs, where rule bases and membership functions are typically designed heuristically and remain fixed. By contrast, FMRLC offers a structured way to continuously tune these parameters during operation, thus enhancing performance and robustness. However, while FMRLC improves adaptability, it still relies on user-defined learning parameters, which are often chosen empirically and may lack strong theoretical guarantees. Moreover, its ability to achieve truly optimal control for general nonlinear systems remains a topic of ongoing research.

4.3.3. Feedback Linearisation with MPC

Feedback Linearisation (FBL) is a nonlinear control strategy that algebraically transforms a nonlinear system into an equivalent linear system through a specific state transformation and feedback control law [41]. This transformation enables the application of linear control techniques to nonlinear systems, simplifying controller design and analysis. However, traditional FBL methods often face challenges in handling input and output constraints. Combining FBL with MPC enables the use of linear MPC strategies. This integration not only allows the handling of constraints but also reduces the computational burden compared to implementing a full nonlinear MPC. A simple representation of this architecture can be seen in Figure (4.6). In [42], a demonstration of this approach can be seen for a flight path following application. It outperforms traditional FBL combined with proportional-PID controllers, which tend to exhibit stronger oscillations in control inputs. However, a significant challenge in this combined approach is the dependency on the accuracy of the feedback linearisation. Any model inaccuracies in the FBL stage can propagate through the system and affect overall performance. Traditionally, precise knowledge of the system model is assumed, which makes the controller susceptible to performance degradation when faced with modelling inaccuracies or disturbances. Recent studies like the one performed by Dutta et al. in [43] have tried to improve on this by making the control strategy adaptive by incorporating mechanisms that dynamically adjust the control strategy in response to real-time discrepancies between the predicted and actual system behaviour.

4.3.4. Reinforcement Learning with MPC

Another hybrid approach is combining reinforcement learning (RL) with model predictive control (MPC). This can take different forms one of which is an RL algorithm operating at a high level and calling an MPC for low-level control. For example, a recent framework integrates Deep RL, MPC, and Graph Neural Networks (GNN) for multi-robot path planning and task allocation [44]. In this DRL-MPC-GNN model, deep

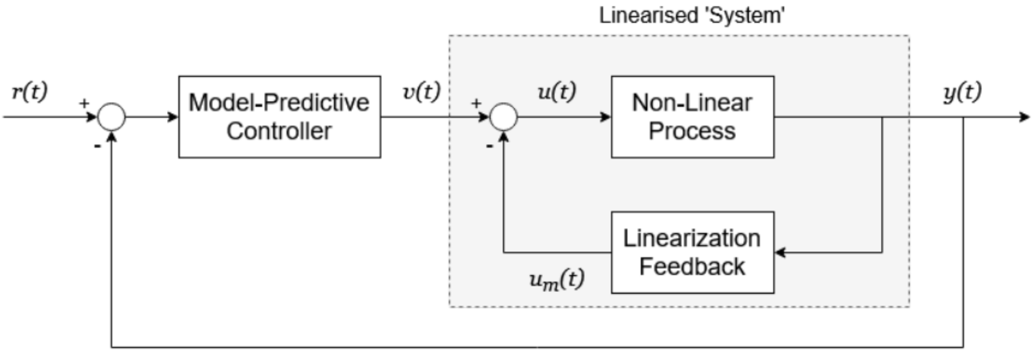


Figure 4.6: FBL-MPC Architecture

RL agents learn a policy for high-level decisions (like which region to search or which task to do next) while a local MPC algorithm computes feasible paths to implement those decisions. The advantage of this integration is that the MPC can correct short-term mistakes of the RL (like if the RL chose a goal slightly too close to another robot, the MPC might still avoid a collision by adjusting the path). Meanwhile, the RL can compensate for the limitations of greedy MPC (which might get stuck in local optima) by occasionally taking exploratory action like sending one robot far out to a new area even if local gain is small, in anticipation of future rewards. Another application involves the use of RL to adjust the horizon and cost weights of an MPC for a multi-robot system in real time to improve energy usage [45]. Figure 4.7 provides a schematic of such an architecture. Thus, the combination of RL with MPC can bring the advantage of adaptability with model-based control. However, such an approach is not trivial, with reward attribution and computation challenges.

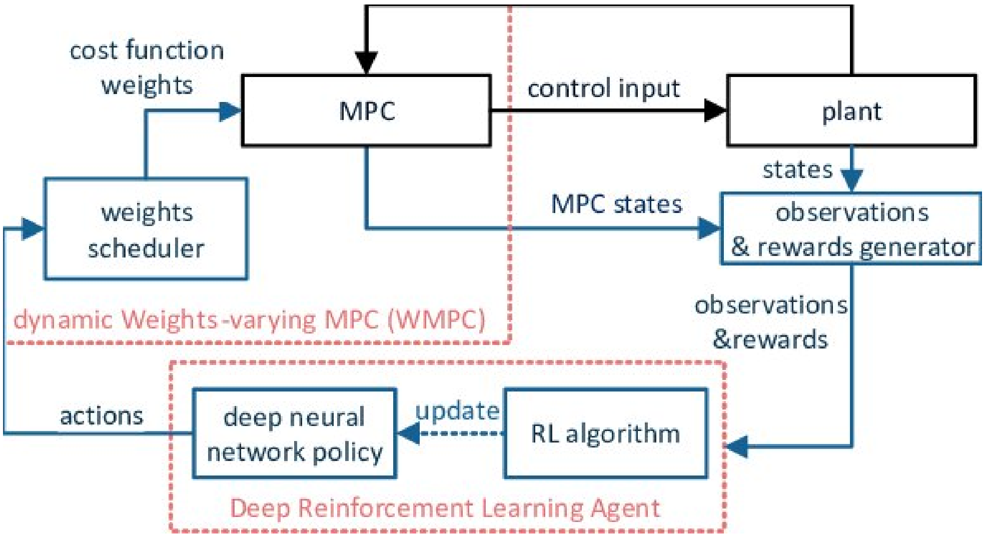

cost function weights
MPC
control input
plant
states
weights scheduler
MPC states
observations & rewards generator
dynamic Weights-varying MPC (WMPC)
observations &rewards
actions
deep neural network policy
update
RL algorithm
Deep Reinforcement Learning Agent

Figure 4.7: RL-MPC Architecture from [46]

5 Search and Rescue Systems

This chapter reviews existing research on Search and Rescue Systems (SaRS). By assessing and comparing the current body of work, it identifies both the latest advancements and ongoing challenges within the field. This is meant to establish the motivation behind the proposed thesis and provides a theoretical foundation for various design decisions made throughout this research.

5.1. Operational Environment

Robotic SaRS must operate effectively across diverse environments, broadly categorised into indoor (built or enclosed spaces) and outdoor (open or natural terrains). Each environment presents distinct challenges, significantly influencing robot design, navigation strategies, and overall effectiveness.

Indoor environments, such as collapsed buildings, typically feature complex and cluttered layouts, comprising intricate arrangements of rooms, interconnected corridors, multiple levels, and substantial debris. These conditions pose practical difficulties for robotic exploration and search operations [47], as movement becomes severely restricted. Ground robots frequently encounter obstacles like debris piles, damaged furniture, and staircases, necessitating robust locomotion capabilities to navigate such challenging terrain effectively. Small UAVs offer agility and can bypass ground obstacles; however, they risk collisions with walls, hanging cables, or other indoor hazards. Flying in confined indoor spaces introduces additional aerodynamic challenges, such as turbulence and limited airflow, demanding sophisticated stabilisation and precise control strategies [48]. Furthermore, limited indoor space restricts battery size, significantly shortening flight endurance and the amount of computational resources robots can carry, directly impacting their capabilities [49]. Another critical challenge is the inherent unpredictability of indoor disaster scenes, which reduces the effectiveness of pre-existing maps or planning strategies based on prior information. This issue is exacerbated by unreliable or absent GPS signals within enclosed spaces, forcing robotic systems to employ intelligent real-time decision-making to safely navigate unknown environments and achieve effective area coverage without collisions [50]. UAVs, particularly rotorcraft, rely heavily on accurate attitude estimation to maintain stability, where even minor positional drifts could lead to collisions with environmental objects [8]. Perception in cluttered and degraded indoor conditions also presents a major challenge. Robotic perception capabilities are significantly impacted by various factors, including lighting variability, visual clutter, occlusions, and inherent sensor limitations in confined spaces. Lighting conditions in indoor disaster scenarios can vary dramatically, ranging from near total darkness to intense glare, with sudden and unpredictable illumination shifts, severely affecting visual sensing performance [51]. To mitigate these perception challenges, integrating multi-modal sensing methods, such as thermal imaging, acoustic sensors, or LiDAR, is beneficial, providing complementary data to purely vision based systems. However, environmental contaminants commonly present during disaster scenarios, such as dust, smoke, and other airborne particles, can considerably degrade sensor performance. Visual sensors may become obscured by smoke and dust, while other sensors may experience interference or reduced accuracy due to such contaminants [52]. Notable applications exploring robotic systems in indoor SaR contexts, though limited, can be found in the works by Sampedro et al. [49], Mano et al. [53], Murphy [54], and Kohlbrecher et al. [55].

Outdoor SaR missions typically take place in expansive environments, which permits the deployment of larger and faster robotic platforms equipped with extended operational range, greater computational capacity, and advanced sensing systems. Unlike cluttered indoor environments, outdoor obstacles and dynamic elements tend to be more sparsely distributed. A significant challenge the outdoor environment poses is maintaining reliable communication over large distances, often in remote regions without established communication infrastructure [56]. Additionally, robotic SaR systems operating outdoors are exposed to varying and sometimes extreme weather conditions, significantly impacting their performance, particularly in perception and localisation tasks [57] [58]. Outdoor SaR operations can occur in urban (USaR), wilderness (WiSaR), and marine (ASR) environments. Each environment presents unique technical challenges and shapes the mission profile distinctly. In USaR scenarios, the primary objective typically involves locating multiple stationary victims within a defined area, though the exact number of victims may not be predetermined. Conversely, WiSaR and ASR scenarios often involve searching for a fixed number of non-stationary or semi stationary targets distributed across considerably larger regions [59]. Moreover, marine environments generally lack structures or obstacles such as buildings, debris, and trees, significantly influencing mission planning strategies. Relevant studies exploring robotic applications in USaR include works by [60], [61], and [62]. WiSaR applications have been examined in [63], [64], and [65].

5.2. Fleet Profile

Having reviewed the different operational environments and their impact on SaR missions, it is clear that there is a need to study different kinds of robotic systems that are available. We mainly discuss land based search and rescue scenarios and hence our focus will be on UAVs and UGVs.

UAVs, with their inherent aerial mobility, provide a unique vantage point for rapid deployment and wide-area surveillance. Their ability to access difficult terrains from an aerial perspective allows for the swift assessment of disaster zones and the identification of potential areas of interest that might be inaccessible to ground-based teams. Largely, UAVs can be of three different classes: Fixed-wing, Rotary and Hybrid. Fixed-wing UAVs are capable of achieving long flight times and demonstrate high energy efficiency, enabling them to cover extensive areas rapidly. They often possess the capacity to carry heavier payloads, allowing for the deployment of more sophisticated sensors or larger batteries for extended missions. This makes them well-suited for initial large-scale search efforts, mapping disaster zones, and conducting environmental monitoring over vast and potentially inaccessible terrains. However, they typically require a runway or a launching mechanism for takeoff and a designated area for landing, restricting their deployment in environments lacking such infrastructure. Additionally, their inability to hover limits their maneuverability in confined spaces and makes close-up inspection of specific points of interest more challenging [66]. Rotary-wing UAVs, encompassing both multirotor and helicopter configurations, are able to tackle some of the limitations of fixed-wing UAVs, due to their ability to take off vertically and hover [67]. Their high maneuverability and agility allow them to navigate through cluttered areas and even operate effectively in both outdoor and indoor settings [66]. Rotary-wing UAVs are commonly used for on-site monitoring, target localisation, indoor exploration of damaged buildings, and even for the delivery of small but critical supplies to stranded individuals. Despite these benefits, rotary-wing UAVs typically exhibit shorter flight times compared to fixed-wing UAVs due to their higher energy consumption required for continuous rotor operation. They also generally have a lower payload capacity, creating a trade-off between the weight of essential sensors and the duration of the mission. In recent times there has been sufficient interest in the use of Hybrid UAVs which combine the strengths of both fixed-wing and rotary-wing platforms. They can take off and land in confined spaces without the need for runways and then transition to efficient horizontal flight for longer-range travel. This makes them highly adaptable to diverse SaR mission phases, potentially enabling rapid initial assessment followed by localised operations. Hybrid UAVs often offer longer endurance than pure rotary-wing models while retaining the flexibility of VTOL. Their applications in SaR include infrastructure inspection, which can be crucial for assessing damage after a disaster, and disaster response scenarios that demand wide area coverage and the ability to carry out detailed operations in specific locations [68]. Current research extensively focuses on utilising UAVs for victim detection, employing multi-sensor payloads like visual and thermal cameras for better perception in various environmental conditions [69] [70]. Beyond victim detection, research also investigates the role of UAVs in supporting broader SaR activities. Studies have explored their utility in WiSaR operations for tasks such as assessing environmental risks, transporting essential equipment to remote locations, and even establishing temporary communication networks in areas where infrastructure has been damaged [6].

There has also been recent research in the application of UGVs in SaR operations. As UGVs offer the capability for ground-level interaction, providing the means to navigate complex and cluttered environments, manipulate objects, and often carry heavier payloads and specialised sensors for detailed investigation and victim assistance [67]. These robots come in various forms, primarily categorised by their mode of locomotion: tracked, wheeled, and legged. Tracked UGVs, also known as crawler robots, utilise continuous tracks for movement, offering traction and stability on uneven and rough terrain, including the rubble and debris commonly encountered in disaster zones [71]. Tracked UGVs are often employed for tasks such as the inspection of collapsed structures, searching for victims within rubble piles, and performing manipulation tasks like the removal of debris to clear pathways or access trapped individuals [72]. However, on flat surfaces, tracked robots can be slower than their wheeled counterparts, and they may experience difficulties with very tight turns in confined spaces. Legged UGVs, which include bipedal, quadrupedal, and other multi-legged configurations offer adaptability to highly complex and unstructured environments, including the ability to navigate stairs and obstacles more effectively than wheeled or tracked robots [73]. Current research and development efforts in SaR are focused on leveraging their unique mobility for tasks requiring navigation through highly unstructured environments and potential interaction with victims [74]. However, the design and control systems for legged robots are typically complex, leading to higher power consumption compared to wheeled robots and potentially lower speed.

Further, in terms of multi-agent SaRS, a differentiation can be made in terms of the homogeneity of agents. Employing homogeneous teams of robots, where all agents possess identical performance capabilities, functionalities, and equipped sensors, offers several potential advantages for Search and Rescue operations. One key benefit is the inherent redundancy such a fleet provides. In the event of a robot failure, any other agent within the team is equally capable of stepping in to perform the required tasks. Homogeneous teams also facilitate task parallelism, where multiple identical robots can be assigned to perform the same type of task concurrently, leading to a faster completion of the overall mission [67]. For instance, a team of UGVs can automate the movement of goods within warehouses or across manufacturing facilities efficiently [75]. On the other hand, the benefit of heterogeneous teams is leveraging of the diverse and complementary capabilities of each robot type. By strategically combining robots with different strengths, a heterogeneous team can address a wider range of challenges encountered in complex disaster scenarios more effectively than a homogeneous team. For example, fast-flying fixed-wing UAVs can quickly survey large areas to identify potential victim locations, while more maneuverable rotary-wing UAVs can conduct closer inspections and access confined spaces. A significant portion of current research on non-homogenous robot teams in SaR focuses on the collaboration between UAVs and UGVs. This synergistic approach recognises the complementary strengths of aerial mobility and ground-based interaction. For example, UAVs can rapidly survey large areas, identify points of interest, and provide real-time imagery, which can then guide UGVs equipped with specialised tools to navigate to those locations on the ground to perform rescue operations or conduct detailed inspections [67]. Studies like [76] have demonstrated that such multimodal robotic search teams, when coordinated effectively, can achieve significantly higher search success rates compared to teams composed of only unimodal or homogeneous robots.

5.3. Perception modes

Situational and perceptual awareness of SaRS agents is essential both for purposes of locomotion and victim search in a disaster scenario [77]. Most research emphasises visual perception methods utilising optical cameras, infrared (heat) sensors, or Light Detection and Ranging (LiDAR) systems. Visual information acquired by robotic agents is especially critical when inconsistencies arise between real-time sensor observations and the environment model maintained by the mission planning system. Furthermore, the integration of acoustic sensors can significantly enhance situational awareness by facilitating the detection and localisation of victims through auditory cues such as vocalisations or distress calls. A practical SaR system incorporating acoustic sensors alongside optical cameras and heat sensors has been explored in [78]. Further studies have examined combining biosensors with optical imaging technologies to identify victims by detecting vital signs, as demonstrated by Cao et al. [79]. An important aspect in relation to perception is sensor noise which leads to perceptual uncertainty. In [63], during field evaluations of their SaRS, the authors reported perceptual noise in the optical camera feeds of the UAVs, which introduced uncertainty in victim identification from captured images. To address this issue, they recommended the deployment of higher-resolution imaging sensors. Further, Koning et al. provided a way to account for the uncertainties in perception due to distance from targets and inherent sensor imperfections in their mathematical formulations, in [80].

5.4. Mission Planning

Mission planning in robotic SaR generally involves two primary tasks: assigning search regions or tasks to robots and determining the best paths for them to follow. Task assignment, known as Multi-Robot Task Allocation (MRTA), becomes especially important when there are more tasks than available robots. Efficient MRTA is crucial for coordinating robots effectively, helping them work towards common goals and optimising overall mission performance.

MRTA must consider numerous factors, such as the complexity and nature of tasks, robot capabilities, environmental conditions, and communication limitations. For example, tasks like aerial surveys are suited to drones (UAVs), whereas tasks involving carrying heavy objects or navigating tight spaces may require ground robots (UGVs) with robotic arms or special mobility features. Leveraging the diverse capabilities of different robot types is essential. Drones offer quick deployment and excellent mobility, while ground robots can carry heavier payloads and perform reliably on uneven terrain.

Environmental constraints—such as obstacles, changing conditions, or areas without GPS—also significantly influence task allocation. Robots operating in GPS-denied environments, for instance, might need advanced localisation technologies like SLAM. MRTA objectives can vary, aiming to minimise completion time, maximise coverage, reduce energy consumption, or achieve a combination of these goals. Allocation strategies can be centralised, decentralised, or hybrid. Centralised approaches involve a central controller that allocates tasks using comprehensive knowledge of the system but can be vulnerable to single points of failure [80]. Decentralised approaches allow robots to independently make decisions, increasing robustness but potentially compromising optimality [81]. Market-based mechanisms are another popular strategy, where robots bid for tasks based on estimated costs and capabilities [82]. Additionally, bio-inspired methods such as ant colony optimisation or particle swarm optimisation have been applied for distributed task allocation in communication-constrained and dynamic environments [83].

Path planning, the second major component, involves finding optimal routes for robots from their current positions to their target destinations. Effective path planning must address obstacle avoidance, dynamic environmental changes, and robot-specific constraints. Classic algorithms like A* and Dijkstra remain popular because they reliably provide optimal solutions in static environments [84].

However, modern SaR scenarios frequently involve dynamic and partially unknown environments, with unpredictable obstacles and hazards. In these cases, MPC has become increasingly useful. MPC predicts future states over a short horizon, allowing robots to adjust their paths in real time. It can optimise for various goals, including energy efficiency, urgency, or safety [85].MPC has also been used in online planning where the environment evolves during the mission. Yao et al. proposed an MPC framework that incorporates collision costs with dynamic and static obstacles into the objective function, along with a control effort term reflecting the energy cost of trajectory changes [86]. This approach, however, relies on deterministic models of obstacle motion, which may not be realistic in unstructured settings. Despite its strengths, MPC has limitations. Its local optimisation horizon can result in suboptimal global paths, particularly in complex or maze-like environments where it may lead the robot into dead ends. To address this, hybrid approaches have been proposed, where global path planning is handled by algorithms like A*, and MPC is used to track the generated path while adjusting for real-time disturbances. For instance, Ishihara et al. introduced a penalty function into the MPC cost formulation to prevent deadlocks by penalising low-speed or idle states, thereby maintaining continuous forward motion [87]. Additionally, in multi-robot scenarios, the local focus of MPC might cause coordination problems or conflicts. Furthermore, solving MPC problems can become computationally intensive in complex environments. To overcome these challenges, MPC is increasingly being used in supervisory or event-triggered roles rather than direct path planning. For instance, Wu et al. developed an event-triggered coordination framework for UAV-UGV teams, where the UAV supervises the UGV's progress and intervenes only under certain conditions [88]. MPC is used not for path generation but to determine when intervention will yield significant improvements, thus optimising mission efficiency while minimising computational and communication overhead.

Recent trends increasingly favour hybrid planning architectures, which combine fast, reactive methods with globally informed decision-making. For example, Koning and Jamshidnejad proposed a hierarchical framework using Fuzzy Logic Controllers (FLCs) at the local level for real-time decision-making and path prioritisation. FLCs use heuristic IF-THEN rules to assign urgency scores to areas within the robot's perception field, which are then evaluated using A* for shortest paths. These candidate paths are graded and selected based on their strategic value for the search mission. At the global level, MPC ensures coordination between robots, avoiding redundant effort by assigning unique search regions to each unit [80]. This hybrid strategy balances real-time adaptability with coordinated global performance and reduces computational load by limiting MPC to supervisory roles while using lightweight heuristics locally. Another application was seen in [89], where a 3-layer hierarchical controller is implemented to coordinate a multi-robot system for search tasks.

5.5. Simulation Design

Ultimately, to evaluate the performance of developed SaRS, it is essential to construct a SaR simulation. Ideally, such testing would involve creating a realistic, life-sized disaster environment paired with physical robotic systems. However, developing this type of high-fidelity physical setup is typically prohibitively expensive and time-consuming. Additionally, when a new system is in its initial stages, virtual simulation is preferable as it allows the concept to be tested safely, ensuring that any system failures remain contained within the virtual domain. Therefore, this research will employ a virtual simulation environment as a test bed, enabling simulated agents to operate within a controlled disaster scenario and execute victim search tasks. A widely recognised platform for such simulations is RoboCup Rescue, which offers a standardised benchmark for evaluating developments in SaRS research. Key simulation components of the RoboCup Rescue framework are detailed in [90]. These include a structural damage model, wherein each building is assigned a *collapse degree* to represent its level of structural compromise. Additionally, buildings possess a *fieryness* attribute, which quantifies fire intensity and associated damage. Victim behaviour is simulated through two primary parameters: *buriedness level*, indicating the victim's mobility, and *damage level*, representing their health status. The simulator also incorporates environmental dynamics by applying a uniform, time-invariant wind vector across the entire environment. Seraji in [91] designed a simulated environment with varying terrain features (like slopes, steps, and roughness). The environment was represented in the form of a grid, with each cell containing sensor-derived features. These features include slope angle, step height (maximum elevation difference) and terrain roughness, based on which a traversability index value is assigned to the cell. This value is considered to be a quantification of how easily a robot can traverse a given terrain region.

In terms of simulation tools, the Robocop Rescue simulator is a common choice for SaRS applications [92]. It is designed to model earthquake-like scenarios and provide flexible 2D map generation at the scale of city blocks. The platform supports the creation of dynamic, multi-agent scenarios, offers standardised performance evaluation frameworks, and is open-source. However, its environmental modelling tools are largely tailored to earthquake disasters, with limited native support for other disaster types such as floods or hurricanes. In contrast, Gazebo is a general-purpose 3D robotic simulation environment with a high-fidelity physics engine, realistic sensor emulation, and seamless integration with ROS [93]. While not originally developed for SaR, its versatility allows researchers to model both indoor and outdoor environments with considerable physical realism, including complex terrains, dynamic obstacles, and multiple heterogeneous agents. Gazebo supports the simulation of wheeled and aerial robots (UGVs and UAVs), making it suitable for evaluating locomotion, perception, and navigation algorithms under realistic conditions. However, it lacks standardised SaR-specific toolboxes, such as victim models or domain-specific performance metrics, which limits its ability to serve as a comprehensive SaR benchmarking platform. Additionally, MATLAB-Gazebo simulators allow verification of robotic behaviours and complex algorithms under dynamic, realistic conditions within Gazebo's realistic physics environment [94]. Similarly, MATLAB's integration with Unity is useful for 3D visualisation and interaction-intensive applications. This includes human–robot interactions, advanced visual feedback, and detailed manipulation scenarios. Both MATLAB–Gazebo and MATLAB–Unity integrations support real-time, bidirectional data exchange.

5.6. Summary of Gaps

Search and rescue robotics is a rapidly evolving field, but it remains a novel area of research with significant opportunities for further exploration. This is particularly true for indoor search and rescue operations, where the body of research is even more limited compared to outdoor environments. Indoor SaR presents unique challenges, such as navigating confined spaces, dealing with poor visibility, and managing complex structures. These challenges differ significantly from those encountered in outdoor environments.

Navigation within complex and confined structures, requires robots to maneuver through intricate layouts having interconnected corridors, multiple levels, narrow passages, and substantial debris. A critical challenge in indoor environments is the absence or severe limitation of Global Positioning System (GPS) signals, which are fundamental to navigation in outdoor settings but largely inaccessible indoors. The absence of reliable GPS compels reliance on alternative methods like IMUs, odometry, LiDAR, and Simultaneous Localisation and Mapping algorithms. However, these alternative methods are prone to drift and inaccuracies, particularly in dynamic or heavily obstructed environments. Additionally, cooperative localisation strategies, which involve multiple robotic agents sharing positional information, remain challenging to implement effectively due to coordination difficulties and communication constraints. Dynamic environmental conditions in the aftermath of disasters involves shifting debris, unstable structures, and frequent environmental changes. Robots must therefore incorporate real-time adaptive capabilities and advanced obstacle avoidance algorithms that allow for effective navigation and operational adjustments to rapidly changing scenarios. Further, robots must differentiate between stable surfaces suitable for traversal and unstable debris. Another unique challenge is visual clutter and occlusions, common in indoor disaster scenes, where scattered debris or structural collapses obstruct clear views. Unlike humans, robots currently struggle to infer complete objects from partially visible data, highlighting gaps in current perception algorithms.

In addressing these challenges, multi-robot systems offer considerable advantages. Such systems can cover more ground, provide redundancy, and perform tasks in parallel, making them well-suited for the demanding conditions of time-sensitive indoor SaR. Moreover, heterogenous systems combined with efficient task allocation algorithms, provide an opportunity to leverage their complementary capabilities, significantly improving mission efficiency and effectiveness. Specifically,systems that combine Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs) offer distinct advantages. The UAV-UGV collaboration enables task sharing, where UAVs can perform rapid aerial reconnaissance while UGVs conduct detailed ground-level searches. Evidence of this can be seen in research by Asadi et al. and Sask et al., where they design and implement a UAV-UGV collaborative system for monitoring and surveillance of indoor spaces [95][96]. In their implementations, the UAV provided rapid aerial survey capability, and aided the UGV in performing detailed, ground level search. However, such system require reliable algorithms and frameworks that dynamically adapt task assignments in response to real-time environmental feedback and maintain effective coordination and communication among multiple robots under constrained conditions. Hierarchical control frameworks are particularly advantageous in such applications because they effectively address the inherent complexity and coordination challenges. A hierarchical structure, like the one proposed by Koning and Jamshidnejad [80], combines the strengths of centralised and decentralised control methods, allowing for both global optimisation and local adaptability. This paper uses the MPC layer in a conflict resolution role, to ensure that the same region is not being explored by multiple robots. Global MPC can also be applied in a task reallocation role. This enables efficient division of labour and resource utilisation, optimising search coverage. For instance, in a UAV-UGV system, the UAV, with its broader aerial perspective and faster movement, could be dynamically re-assigned to quickly survey larger areas or regions with higher target probabilities. Meanwhile, the UGV could focus on thoroughly inspecting areas identified by the UAV, leveraging its ability to navigate complex terrain and carry specialised sensors. This dynamic task allocation helps maximise both area coverage and victim detection efficiency, ultimately leading to faster and more successful search operations. It also allows the system to compensate for individual robot limitations and capitalise on their strengths. For instance, if a UAV detects a potential target, the system can immediately re-task a nearby UGV to confirm the target's identity [65]. By employing a global MPC for high-level task allocation, the system can optimise overall performance, taking into account global objectives and constraints, while delegating low-level control to computationally efficient, heuristic FLC controllers. This combination is beneficial as FLC, with its ability to mimic human-like reasoning, can effectively handle local uncertainties and dynamic obstacles in real time, while the MPC ensures global coordination and efficient task distribution. Despite the evident advantages, there is insufficient research exploring a similar hierarchical architecture in SaR operations. The Koning paper provides a promising proof of concept for a hierarchical decision-making architecture in heterogeneous multi-robot systems for environmental coverage and target identification. However, there remains room to explore alternate frameworks and also to extend the environment model to better reflect the complexities and challenges of real-world scenarios. Hence, for the above-mentioned reasons, there is sufficient motivation to design a hierarchical MPC-FLC decision-making framework for a UAV-UGV system, to maximise coverage of an unknown environment and rapid victim identification, during a SaR situation.

6 Thesis Framework

The objective of the literature review was to review state-of-the-art research in SaRS, identify a research gap, and to propose a solution to address the gap. In this section, a formal thesis plan is described.

6.1. Problem Definition

This thesis aims to design and implement a decision-making architecture for a collaborative, multi-agent UAV-UGV system. The aim is to optimise coverage and exploration of an unknown environment while maximising the time efficiency of detecting victims.

6.2. Proposed Methodology

To achieve this, a hierarchical control system will be implemented. Both the UAV and UGV have a decentralised, local fuzzy logic controller, which is used to handle sensor data, characterise cells, search for shortest paths passing through high-priority regions, and execute these paths. The FLC controller will use a Mamdani structure. The UAV adopts a rapid search behaviour (faster coverage speed and wider perception field) and carries out a broad sweep. It identifies victims and regions requiring UGV intervention. Simultaneously, the UGV carries out its search behaviour, identifying victims and regions requiring UAV intervention. The centralised global MPC controller continuously monitors the evolving environmental map and intervenes only to reallocate tasks among the two agents based on the latest information. This intervention would be based on the limitation the environment condition imposes on the agents and the relative advantage an agent has in executing the search task in a certain set of conditions. A custom simulation environment will be used to simulate the approach, using MATLAB to implement the decision making algorithm and combining it with a 3D environment simulator.

6.3. Research Questions

The main research question for this thesis can be formulated as:

> Can an adaptive, hierarchical MPC-FLC control framework for a collaborative UAV-UGV multi-agent team improve the efficiency of an environment constrained and dynamic SaR mission, in terms of area coverage and victim search detection?

The main research question will be addressed by evaluating the following sub-questions.

1. How should the environment be designed to simulate real-world operational challenges in an indoor SaR scenario?
   (a) Which primary static and dynamic elements must be modelled?
   (b) How do the individual elements impact the operations (perception and movement) of UAVs and UGVs?
   (c) How should the agents be modelled?
   (d) How should experiments and specific scenarios be designed for fair benchmarking with state-of-the-art methods?
2. How can the MPC-FLC framework be applied to the mission planning problem?

(a) How can the environment be characterised for the purpose of priority assignment and evaluation of multiple tasks?

(b) How are the specific capabilities of UAV and UGV agents incorporated in the FLC controller?

(c) How should the fuzzy rules be defined?

(d) What should be the motivation or trigger for MPC to intervene and reallocate tasks?

(e) How should the optimisation problem to achieve efficient task allocation?

(f) What should the information exchange framework be between the agents and the supervisory controller?

3. What are the performance criteria for fare comparison to state-of-the-art approaches in the same problem scenario?

   (a) Which criteria should be used to evaluate performance in terms of victim detection and area coverage? (Victim detection time, Number of victims detected, Rise term of the certainty of knowledge about the environment or certainty evolution)

   (b) How should the task allocation efficiency and impact be evaluated?

References

[1] Hannah Ritchie et al. “Natural Disasters”. In: *Our World in Data* (2022). https://ourworldindata.org/natural-disasters.

[2] United Nations Office for Disaster Risk Reduction (UNDRR). *UNDRR Terminology on Disaster Risk Reduction*. Accessed: 2025-03-24. 2024. URL: `https://www.undrr.org/drr-glossary/terminology`.

[3] Amir Khorram-Manesh et al. *Handbook of Disaster and Emergency Management (2nd Edition, 2021)*. Oct. 2021. DOI: `10.5281/zenodo.5553076`.

[4] Jeffrey Delmerico et al. “The current state and future outlook of rescue robotics”. In: *Journal of Field Robotics* 36.7 (2019), pp. 1171–1191.

[5] Dorothy Ryan. “Technology Confronts Disasters”. In: *MIT News* (2015). URL: `https://news.mit.edu/2015/technology-confronts-disasters-1020`.

[6] Craig Vincent-Lambert et al. “Use of unmanned aerial vehicles in wilderness search and rescue operations: A scoping review”. In: *Wilderness & Environmental Medicine* 34.4 (2023), pp. 580–588.

[7] Jake N McRae et al. “Using an unmanned aircraft system (drone) to conduct a complex high altitude search and rescue operation: a case study”. In: *Wilderness & environmental medicine* 30.3 (2019), pp. 287–290.

[8] Guido Croon et al. “Challenges of Autonomous Flight in Indoor Environments”. In: Oct. 2018, pp. 1003–1009. DOI: `10.1109/IROS.2018.8593704`.

[9] David Paez et al. “Distributed Particle Swarm Optimization for Multi-Robot System in Search and Rescue Operations”. In: *IFAC-PapersOnLine* 54.4 (2021). 4th IFAC Conference on Embedded Systems, Computational Intelligence and Telematics in Control CESCIT 2021, pp. 1–6. DOI: `https://doi.org/10.1016/j.ifacol.2021.10.001`. URL: `https://www.sciencedirect.com/science/article/pii/S2405896321014038`.

[10] Ali Agha et al. “Nebula: Quest for robotic autonomy in challenging environments; team costar at the darpa subterranean challenge”. In: *arXiv preprint arXiv:2103.11470* (2021).

[11] Jianqiang Li et al. “Energy-Efficient Ground Traversability Mapping Based on UAV-UGV Collaborative System”. In: *IEEE Transactions on Green Communications and Networking* PP (Aug. 2021), pp. 1–1. DOI: `10.1109/TGCN.2021.3107291`.

[12] Ivana Kruijff-Korbayova et al. “Deployment of Ground and Aerial Robots in Earthquake-Struck Amatrice in Italy (brief report)”. In: Oct. 2016. DOI: `10.1109/SSRR.2016.7784314`.

[13] L.A. Zadeh. “Fuzzy sets”. In: *Information and Control* 8.3 (1965), pp. 338–353. DOI: `https://doi.org/10.1016/S0019-9958(65)90241-X`. URL: `https://www.sciencedirect.com/science/article/pii/S001999586590241X`.

[14] LA Zadeh. “Fuzzy sets for intelligent systems”. In: *Fuzzy Sets, chapitre* 2 (1993), pp. 27–64.

[15] Assefinew Wondosen et al. “Fuzzy Logic Controller Design for Mobile Robot Outdoor Navigation”. In: *arXiv preprint arXiv:2401.01756* (2024).

[16] E.H. Mamdani. “Application of fuzzy algorithms for control of simple dynamic plant”. In: *Proceedings of the Institution of Electrical Engineers* 121 (12 1974), pp. 1585–1588. DOI: `10.1049/piee.1974.0328`. eprint: `https://digital-library.theiet.org/doi/pdf/10.1049/piee.1974.0328`. URL: `https://digital-library.theiet.org/doi/abs/10.1049/piee.1974.0328`.

[17] Tomohiro Takagi et al. “Fuzzy identification of systems and its applications to modeling and control”. In: *IEEE Transactions on Systems, Man, and Cybernetics* SMC-15.1 (1985), pp. 116–132. DOI: 10.1109/TSMC.1985.6313399.

[18] Omar Rodríguez-Abreo et al. “Fuzzy logic controller for UAV with gains optimized via genetic algorithm”. In: *Heliyon* 10.4 (Feb. 2024), e26363. DOI: 10.1016/j.heliyon.2024.e26363.

[19] Adamu Babawuro et al. “INTELLIGENT TEMPERATURE AND HUMIDITY CONTROLLER FOR YAM TUBERS POST-HARVEST STORAGE SYSTEM”. In: *nternational Journal of Mechanical and Production Engineering (IJMPE)* (Jan. 2015), pp. 2320–2092.

[20] Ahmad Din et al. “Behavior-based swarm robotic search and rescue using fuzzy controller”. In: *Computers & Electrical Engineering* 70 (2018), pp. 53–65.

[21] A. J. van der Wal. “Application of fuzzy logic control in industry”. In: *Fuzzy Sets Syst.* 74.1 (Aug. 1995), pp. 33–41. DOI: 10.1016/0165-0114(95)00033-H. URL: https://doi-org.tudelft.idm.oclc.org/10.1016/0165-0114(95)00033-H.

[22] Mohammed Rabah et al. “Autonomous Moving Target-Tracking for a UAV Quadcopter Based on Fuzzy-PI”. In: *IEEE Access* 7 (2019), pp. 38407–38419. DOI: 10.1109/ACCESS.2019.2906345.

[23] Athanasios Tsalatsanis et al. “Dynamic Task Allocation in Cooperative Robot Teams”. In: *International Journal of Advanced Robotic Systems* 6.4 (2009), p. 30. DOI: 10.5772/10678. eprint: https://doi.org/10.5772/10678. URL: https://doi.org/10.5772/10678.

[24] Izzat Aldarraji et al. “Takagi-sugeno fuzzy modeling and control for effective robotic manipulator motion”. In: *arXiv preprint arXiv:2112.03006* (2021).

[25] Yugang Liu et al. “Multirobot Cooperative Learning for Semiautonomous Control in Urban Search and Rescue Applications”. In: *Journal of Field Robotics* 33.4 (2016), pp. 512–536. DOI: https://doi.org/10.1002/rob.21597. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21597. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21597.

[26] Farzad Niroui et al. “Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments”. In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 610–617.

[27] Gabriele Calzolari et al. “Reinforcement Learning Driven Multi-Robot Exploration via Explicit Communication and Density-Based Frontier Search”. In: *arXiv preprint arXiv:2412.20049* (2024).

[28] Yihe Zhou et al. “Is centralized training with decentralized execution framework centralized enough for MARL?” In: *arXiv preprint arXiv:2305.17352* (2023).

[29] Muthumeena Muthiah et al. “Multi robot path planning and path coordination using genetic algorithms”. In: *Proceedings of the 2017 ACM Southeast Conference*. 2017, pp. 112–119.

[30] Samira Hayat et al. “Multi-objective drone path planning for search and rescue with quality-of-service requirements”. In: *Autonomous Robots* 44.7 (July 2020), pp. 1183–1198. DOI: 10.1007/s10514-020-09926-9.

[31] Liwei Yang et al. “Path Planning Technique for Mobile Robots: A Review”. In: *Machines* 11.10 (2023). DOI: 10.3390/machines11100980. URL: https://www.mdpi.com/2075-1702/11/10/980.

[32] Liwei Yang et al. “Path Planning Technique for Mobile Robots: A Review”. In: *Machines* 11.10 (2023). DOI: 10.3390/machines11100980. URL: https://www.mdpi.com/2075-1702/11/10/980.

[33] Carlos Bordons et al. “Model Predictive Control Fundamentals”. In: *Model Predictive Control of Microgrids*. Cham: Springer International Publishing, 2020, pp. 25–44. DOI: 10.1007/978-3-030-24570-2_2. URL: https://doi.org/10.1007/978-3-030-24570-2_2.

[34] Ian McInerney et al. “Towards a Framework for Nonlinear Predictive Control using Derivative-Free Optimization”. In: *IFAC-PapersOnLine* 54.6 (2021), pp. 284–289. DOI: 10.1016/j.ifacol.2021.08.558. URL: http://dx.doi.org/10.1016/j.ifacol.2021.08.558.

[35] Virginia Torczon. “On the Convergence of Pattern Search Algorithms”. In: *SIAM Journal on Optimization* 7.1 (1997), pp. 1–25. DOI: 10.1137/S1052623493250780. eprint: https://doi.org/10.1137/S1052623493250780. URL: https://doi.org/10.1137/S1052623493250780.

[36] D.D. Šiljak. “Decentralized control and computations: status and prospects”. In: *Annual Reviews in Control* 20 (1996), pp. 131–141. DOI: https://doi.org/10.1016/S1367-5788(97)00011-4. URL: https://www.sciencedirect.com/science/article/pii/S1367578897000114.

[37] Mathias Bürger et al. “From non-cooperative to cooperative distributed MPC: A simplicial approximation perspective”. In: *2013 European Control Conference (ECC)*. IEEE. 2013, pp. 2795–2800.

[38] Y.L. Huang et al. “Fuzzy model predictive control”. In: *IEEE Transactions on Fuzzy Systems* 8.6 (2000), pp. 665–678. DOI: 10.1109/91.890326.

[39] Peng Yao et al. “Multi-UAV information fusion and cooperative trajectory optimization in target search”. In: *IEEE Systems Journal* 16.3 (2021), pp. 4325–4333.

[40] J. R. Layne et al. “Fuzzy Model Reference Learning Control”. In: *Advances in Fuzzy Control*. Ed. by Dimiter Driankov et al. Heidelberg: Physica-Verlag HD, 1998, pp. 263–282. DOI: 10.1007/978-3-7908-1886-4_10. URL: https://doi.org/10.1007/978-3-7908-1886-4_10.

[41] Yong-Lin Kuo et al. “Continuous-Time Nonlinear Model Predictive Tracking Control with Input Constraints Using Feedback Linearization”. In: *Applied Sciences* 12.10 (2022). DOI: 10.3390/app12105016. URL: https://www.mdpi.com/2076-3417/12/10/5016.

[42] W. Soest et al. “Combined Feedback Linearization and Constrained Model Predictive Control for Entry Flight”. In: *Journal of Guidance Control and Dynamics - J GUID CONTROL DYNAM* 29 (Mar. 2006), pp. 427–434. DOI: 10.2514/1.14511.

[43] Lakshmi Dutta et al. “An adaptive feedback linearized model predictive controller design for a nonlinear multi-input multi-output system”. In: *International Journal of Adaptive Control and Signal Processing* 35.6 (2021), pp. 991–1016. DOI: https://doi.org/10.1002/acs.3239. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/acs.3239. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/acs.3239.

[44] Zhixian Li et al. “Deep reinforcement learning path planning and task allocation for multi-robot collaboration”. In: *Alexandria Engineering Journal* 109 (2024), pp. 408–423. DOI: https://doi.org/10.1016/j.aej.2024.08.102. URL: https://www.sciencedirect.com/science/article/pii/S1110016824010007.

[45] Baha Zarrouki et al. “A safe reinforcement learning driven weights-varying model predictive control for autonomous vehicle motion control”. In: *2024 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2024, pp. 1401–1408.

[46] Baha Zarrouki et al. “Weights-varying MPC for Autonomous Vehicle Guidance: a Deep Reinforcement Learning Approach”. In: June 2021, pp. 119–125. DOI: 10.23919/ECC54610.2021.9655042.

[47] Fabio Maresca et al. “REACT: Multi Robot Energy-Aware Orchestrator for Indoor Search and Rescue Critical Tasks”. In: *arXiv preprint arXiv:2503.05904* (2025).

[48] Thomas Martin et al. “Flying in air ducts”. In: *arXiv preprint arXiv:2410.08379* (2024).

[49] Carlos Sampedro et al. “A Fully-Autonomous Aerial Robot for Search and Rescue Applications in Indoor Environments using Learning-Based Techniques”. In: *Journal of Intelligent and Robotic Systems* 95.2 (July 2018), pp. 601–627. DOI: 10.1007/s10846-018-0898-1.

[50] Mingyang Lyu et al. “Unmanned Aerial Vehicles for Search and Rescue: A Survey”. In: *Remote Sensing* 15.13 (2023). DOI: 10.3390/rs15133266. URL: https://www.mdpi.com/2072-4292/15/13/3266.

[51] T. Ran et al. “Scene perception based visual navigation of mobile robot in indoor environment”. In: *ISA Transactions* 109 (Mar. 2021), pp. 389–400. DOI: 10.1016/j.isatra.2020.10.023.

[52] S. Solmaz et al. “Robust Robotic Search and Rescue in Harsh Environments: An Example and Open Challenges”. In: June 2024, pp. 1–8. DOI: 10.1109/ROSE62198.2024.10591144.

[53] Hayato Mano et al. “Treaded control system for rescue robots in indoor environment”. In: *2008 IEEE International Conference on Robotics and Biomimetics*. 2009, pp. 1836–1843. DOI: 10.1109/ROBIO.2009.4913281.

[54] Robin R Murphy. “Trial by fire [rescue robots]”. In: *IEEE Robotics & Automation Magazine* 11.3 (2004), pp. 50–61.

[55] Stefan Kohlbrecher et al. “Towards highly reliable autonomy for urban search and rescue robots”. In: *RoboCup 2014: Robot World Cup XVIII 18*. Springer. 2015, pp. 118–129.

[56] Cuebong Wong et al. “Autonomous robots for harsh environments: a holistic overview of current solutions and ongoing challenges”. In: *Systems Science & Control Engineering* 6.1 (2018), pp. 213–219. DOI: 10.1080/21642583.2018.1477634. eprint: https://doi.org/10.1080/21642583.2018.1477634. URL: https://doi.org/10.1080/21642583.2018.1477634.

[57] Hanan Atiyah et al. “Outdoor Localization for a Mobile Robot under Different Weather Conditions Using a Deep Learning Algorithm”. In: *Journal Européen des Systèmes Automatisés* 56 (Feb. 2023), pp. 1–9. DOI: 10.18280/jesa.560101.

[58] Hiroaki Katsura et al. “A View-Based Outdoor Navigation Using Object Recognition Robust to Changes of Weather and Seasons”. In: *Journal of the Robotics Society of Japan* 23 (Jan. 2005), pp. 75–83. DOI: 10.7210/jrsj.23.75.

[59] Sean Grogan et al. “The use of unmanned aerial vehicles and drones in search and rescue operations – a survey”. In: Sept. 2018.

[60] Ross Arnold et al. “Heterogeneous UAV Multi-Role Swarming Behaviors for Search and Rescue”. In: Aug. 2020, pp. 122–128. DOI: 10.1109/CogSIMA49017.2020.9215994.

[61] Zoltán Beck et al. “Online planning for collaborative search and rescue by heterogeneous robot teams”. In: (2016).

[62] Wing-Yue Geoffrey Louie et al. “A victim identification methodology for rescue robots operating in cluttered USAR environments”. In: *Advanced Robotics* 27.5 (2013), pp. 373–384. DOI: 10.1080/01691864.2013.763743. eprint: https://doi.org/10.1080/01691864.2013.763743. URL: https://doi.org/10.1080/01691864.2013.763743.

[63] Lanny Lin et al. “Supporting Wilderness Search and Rescue with Integrated Intelligence: Autonomy and Information at the Right Time and the Right Place”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 24.1 (July 2010), pp. 1542–1547. DOI: 10.1609/aaai.v24i1.7573.

[64] Michael Goodrich et al. “Supporting wilderness search and rescue using a camera-equipped mini UAV: Research Articles”. In: *Journal of Field Robotics* 25 (Feb. 2008), pp. 89–110. DOI: 10.1002/rob.20226.

[65] Zendai Kashino et al. “Aerial Wilderness Search and Rescue with Ground Support”. In: *Journal of Intelligent and Robotic Systems* 99.1 (Nov. 2019), pp. 147–163. DOI: 10.1007/s10846-019-01105-y.

[66] Syed Agha Hassnain Mohsan et al. “Unmanned aerial vehicles (UAVs): practical aspects, applications, open challenges, security issues, and future trends”. In: *Intelligent Service Robotics* (Jan. 2023). DOI: 10.1007/s11370-022-00452-4.

[67] Isuru Munasinghe et al. “A Comprehensive Review of UAV-UGV Collaboration: Advancements and Challenges”. In: *Journal of Sensor and Actuator Networks* 13.6 (2024). DOI: 10.3390/jsan13060081. URL: https://www.mdpi.com/2224-2708/13/6/81.

[68] Adnan Saeed et al. “A survey of hybrid Unmanned Aerial Vehicles”. In: *Progress in Aerospace Sciences* (Mar. 2018). DOI: 10.1016/j.paerosci.2018.03.007.

[69] Fabio Augusto de Alcantara Andrade et al. “Autonomous unmanned aerial vehicles in search and rescue missions using real-time cooperative model predictive control”. In: *Sensors* 19.19 (2019), p. 4067.

[70] Abhijith Valsan et al. “Unmanned Aerial Vehicle for Search and Rescue Mission”. In: June 2020, pp. 684–687. DOI: 10.1109/ICOEI48184.2020.9143062.

[71] Dawn Tilbury et al. “A new breed of robots that drive themselves”. In: *Mechanical Engineering* 133.02 (2011), pp. 28–33.

[72] Geert De Cubber et al. “Search and Rescue robots developed by the European ICARUS project”. In: Oct. 2013.

[73] C. Dario Bellicoso et al. “Advances in real-world applications for legged robots”. In: *Journal of Field Robotics* 35.8 (2018), pp. 1311–1326. DOI: `https://doi.org/10.1002/rob.21839`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21839`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21839`.

[74] Jeffrey Delmerico et al. “The current state and future outlook of rescue robotics”. In: *Journal of Field Robotics* 36.7 (2019), pp. 1171–1191. DOI: `https://doi.org/10.1002/rob.21887`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.21887`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21887`.

[75] Peter R Wurman et al. “Coordinating hundreds of cooperative, autonomous vehicles in warehouses”. In: *AI magazine* 29.1 (2008), pp. 9–9.

[76] Shan Yu Ku et al. “Wilderness Search for Lost Persons Using a Multimodal Aerial-Terrestrial Robot Team”. In: *Robotics* 11.3 (2022). DOI: `10.3390/robotics11030064`. URL: `https://www.mdpi.com/2218-6581/11/3/64`.

[77] J. Casper et al. “Human-robot interactions during the robot-assisted urban search and rescue response at the World Trade Center”. In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 33.3 (2003), pp. 367–385. DOI: `10.1109/TSMCB.2003.811794`.

[78] Subramanium Ganesan et al. “Small disaster relief robots with swarm intelligence routing”. In: *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief*. ACWR ’11. Amritapuri, Kollam, Kerala, India: Association for Computing Machinery, 2011, pp. 123–127. DOI: `10.1145/2185216.2185257`. URL: `https://doi-org.tudelft.idm.oclc.org/10.1145/2185216.2185257`.

[79] Yusen Cao et al. “Mission chain driven unmanned aerial vehicle swarms cooperation for the search and rescue of outdoor injured human targets”. In: *Drones* 6.6 (2022), p. 138.

[80] Christopher de Koning et al. “Hierarchical Integration of Model Predictive and Fuzzy Logic Control for Combined Coverage and Target-Oriented Search-and-Rescue via Robots with Imperfect Sensors”. In: *Journal of Intelligent and Robotic Systems* 107.3 (Mar. 2023). DOI: `10.1007/s10846-023-01833-2`.

[81] Jorge Peña Queralta et al. “Collaborative Multi-Robot Search and Rescue: Planning, Coordination, Perception, and Active Vision”. In: *IEEE Access* 8 (Jan. 2020), pp. 191617–191643. DOI: `10.1109/ACCESS.2020.3030190`.

[82] Jian Tang et al. “Using auction-based task allocation scheme for simulation optimization of search and rescue in disaster relief”. In: *Simulation Modelling Practice and Theory* 82 (2018), pp. 132–146. DOI: `https://doi.org/10.1016/j.simpat.2017.12.014`. URL: `https://www.sciencedirect.com/science/article/pii/S1569190X17301934`.

[83] Heba Kurdi et al. “Bio-Inspired Algorithm for Task Allocation in Multi-UAV Search and Rescue Missions”. In: *AIAA Guidance, Navigation, and Control Conference*. DOI: `10.2514/6.2016-1377`. eprint: `https://arc.aiaa.org/doi/pdf/10.2514/6.2016-1377`. URL: `https://arc.aiaa.org/doi/abs/10.2514/6.2016-1377`.

[84] Xia Chen et al. “Dynamic Path Planning of the UAV Avoiding Static and Moving Obstacles”. In: *Journal of Intelligent & Robotic Systems* 99 (Sept. 2020). DOI: `10.1007/s10846-020-01151-x`.

[85] Jonghoek Kim. “Time-efficient path planning using two virtual robots”. In: *International Journal of Advanced Robotic Systems* 16.6 (2019), p. 1729881419886742. DOI: `10.1177/1729881419886742`. eprint: `https://doi.org/10.1177/1729881419886742`. URL: `https://doi.org/10.1177/1729881419886742`.

[86] Ravi Raj et al. “Discussion on different controllers used for the navigation of mobile robot”. In: *International Journal of Electronics and Telecommunications* 70 (2024).

[87] Shinji Ishihara et al. “A Proposal of Path Planning for Robots in Warehouses by Model Predictive Control without Using Global Paths”. In: *IFAC-PapersOnLine* 55.37 (2022). 2nd Modeling, Estimation and Control Conference MECC 2022, pp. 573–578. DOI: `https://doi.org/10.1016/j.ifacol.2022.11.244`. URL: `https://www.sciencedirect.com/science/article/pii/S2405896322028877`.

[88] Wu Wang et al. “Event-Triggered Intervention Framework for UAV-UGV Coordination Systems”. In: *Machines* 9.12 (2021). DOI: `10.3390/machines9120371`. URL: `https://www.mdpi.com/2075-1702/9/12/371`.

[89] S. Dilip Kumar. “Hierarchical Control for Multi-Robot Systems using Contracts”. Master’s thesis. MA thesis. Delft University of Technology, Oct. 2024. URL: `http://resolver.tudelft.nl/uuid:79e2ae59-751b-467a-814d-46c1ce52f61d`.

[90] Hiroaki Kitano et al. “RoboCup Rescue: Search and Rescue in Large-Scale Disasters as a Domain for Autonomous Agents Research”. In: (Aug. 1999).

[91] Homayoun Seraji. “New Traversability Indices and Traversability Grid for Integrated Sensor/Map-Based Navigation”. In: *Journal of Robotic Systems* 20.3 (2003), pp. 121–134. DOI: `https://doi.org/10.1002/rob.10074`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/rob.10074`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.10074`.

[92] RoboCup Federation. *RoboCup Rescue Simulation Documentation*. Accessed: 2025-03-24. URL: `https://rescuesim.robocup.org/resources/documentation/`.

[93] Open Robotics. *Gazebo Simulator*. `http://gazebosim.org/`. Accessed: 2025-03-24.

[94] MathWorks. *How Gazebo Simulation for Robotics Works*. `https://nl.mathworks.com/help/robotics/ug/how-gazebo-simulation-for-robotics-works.html`. Accessed: 24-Mar-2024. 2024.

[95] Khashayar Asadi et al. “An integrated UGV-UAV system for construction site data collection”. In: *Automation in Construction* 112 (2020), p. 103068. DOI: `https://doi.org/10.1016/j.autcon.2019.103068`. URL: `https://www.sciencedirect.com/science/article/pii/S0926580519306545`.

[96] Martin Saska et al. “Cooperative μUAV-UGV autonomous indoor surveillance”. In: *International Multi-Conference on Systems, Signals & Devices*. 2012, pp. 1–6. DOI: `10.1109/SSD.2012.6198051`.

Part III

Appendix

A Appendix

The MATLAB program used to carry out the simulations detailed in this research has been uploaded to a public GitHub repository (https://github.com/Shubham2001-33/UAV-UGV-Cooperative-Decision-Making-in-SaR). This section contains an overview of the code, how to run it and finally a set of flowcharts for visualising the process flow.

A.1. Requirements

The implementation of this research was carried out in MATLAB R2023a on a PC with AMD Ryzen 7 Processor with 3.2GHz frequency. The simulation makes use of the Global Optimisation Toolbox, and the Parallel Computing Toolbox.

A.2. Code Overview

The main execution file is `main.m`. Upon execution, the user is presented with two dialog boxes to determine the search approach and the simulation scenario. The choice of simulation scenario leads to the calling of the relevant simulation environment function (`randomenv.m`,`case1env.m`,`case2env.m`,`case3env.m`), via the function handle, `init_simenv.m`.

The environment initialisation function randomly constructs the environment map with the relevant environmental parameters via global maps for victim distribution (`M_VIC`), obstacle distribution (`M_OCC`), observability (`M_OBS`), and terrain index(`M_TERR`). It also generates empty containers for performance indicators like the scan certainty map (`M_SCAN`). Further, SaR agents are generated with a set of fixed characteristics, as well as variable parameters, by calling the function `gen_agent.m`. These parameters include the agent type (UAV or UGV), perception radius $r^{\mathrm{p}}_{i,j}$, occlusion sensitivity factor $\alpha_{i,j}$, perception field $E^{\mathrm{p}}_{i,j}$ and a list of identified victims.

Following this, the `simulate_step.m` function is executed, which carries out the system evolution over one time step. In each execution cycle, the victim positions are updated (since the victim position is assumed to be dynamic in the problem formulation) via the `vicpos_update.m` function. The `perception_update.m` function identifies all the cells in the agent's perception field, updates the agent's maps with information about the cells and updates their scan certainty values. Further, the `localprio_update.m` function computes the priority of visitation (ρ), for all cells in ($E^{\mathrm{p}}_{i,j}$). For all these cells, a total of K shortest paths are identified using `plan_kshortest.m`. Next, these paths are graded based on the criteria defined in this thesis in `path_grading.m` and the path with the highest grade is added to a container, P_0, as the local solution.

This is followed by a check to see if there is an **'agent-environment'** conflict, i.e, the mean of observability values for all cells in the perception field of UAVs is less than (`o_thresh`) or the mean of terrain index values for all cells in the perception field of UGVs is greater than (`t_thresh`). If either of the conditions is met, the supervisory controller defined in `MPC_optimise.m` takes over and generates an optimised path, P. This path is generated using a `patternsearch` optimisation function.

The functions `Ant_search.m` and `exhaustive_search.m` execute the ACS and random search approaches. While the pure-MPC approach is executed using the same program as the proposed cooperative method (discussed above), without the warm-start with the set of local paths. Instead, the function `init_path.m` generates a random initial solution for the optimisation function, `MPC_optimise.m`.

A.3. Flowcharts

This section provides flowcharts visualising the process flow of the simulation for the proposed method.

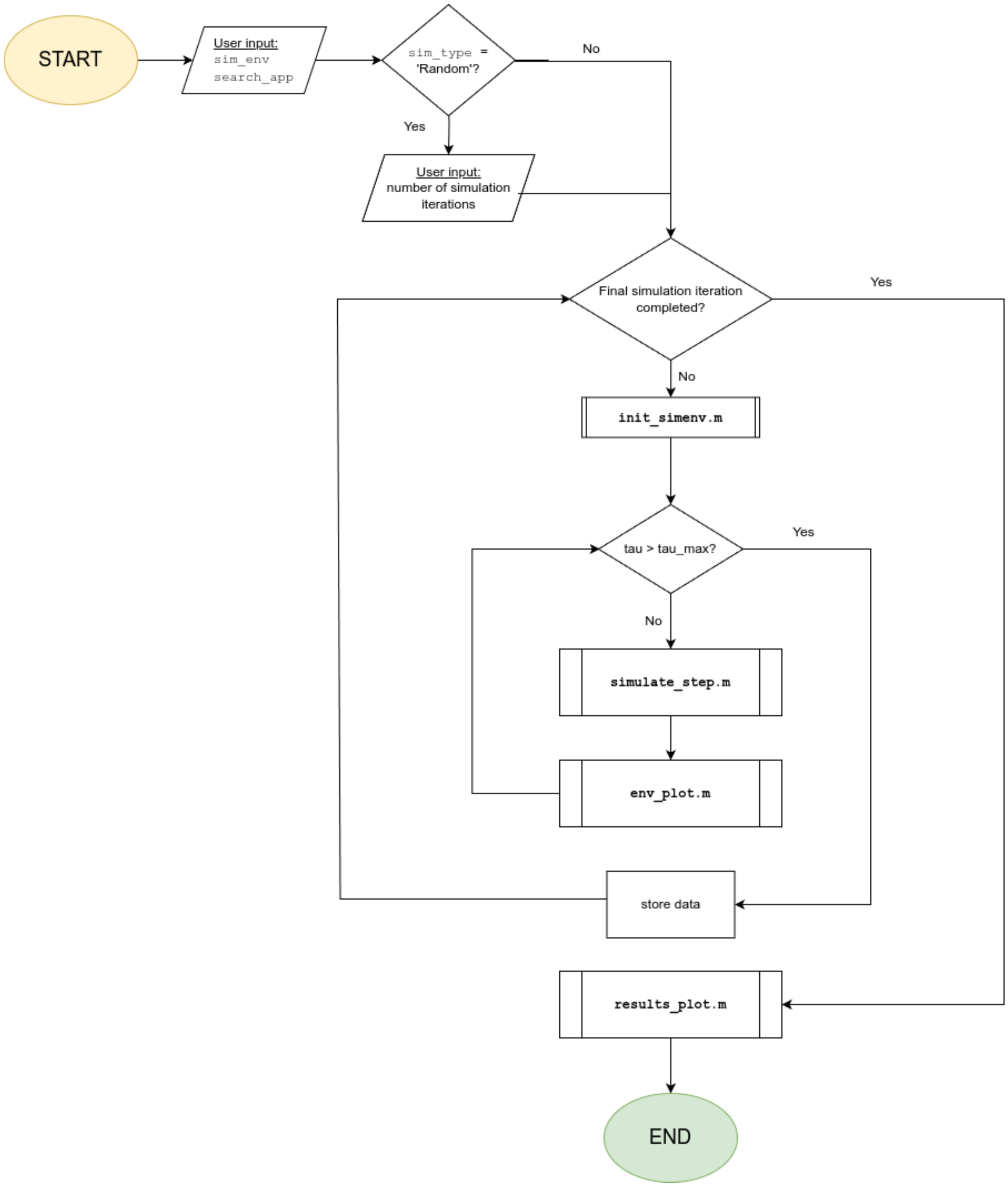


Figure A.1: Flowchart for run file `main.m`

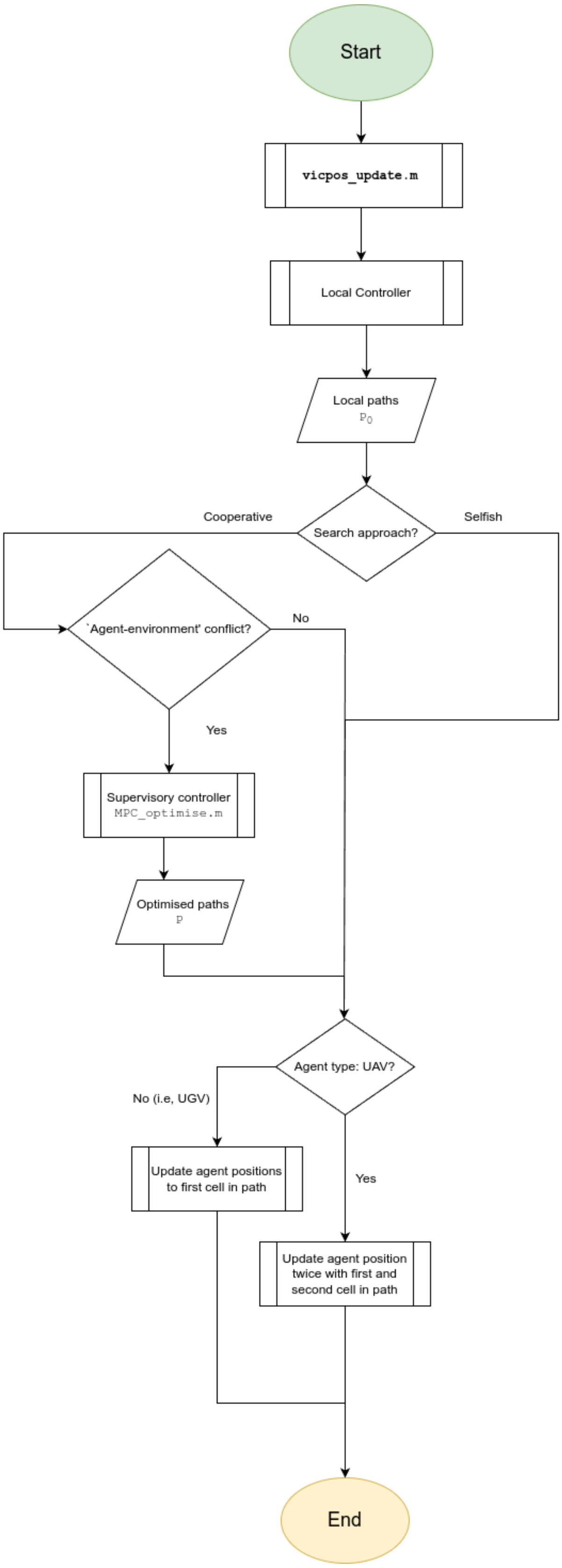


Figure A.2: Flowchart for control execution function `simulate_step.m`

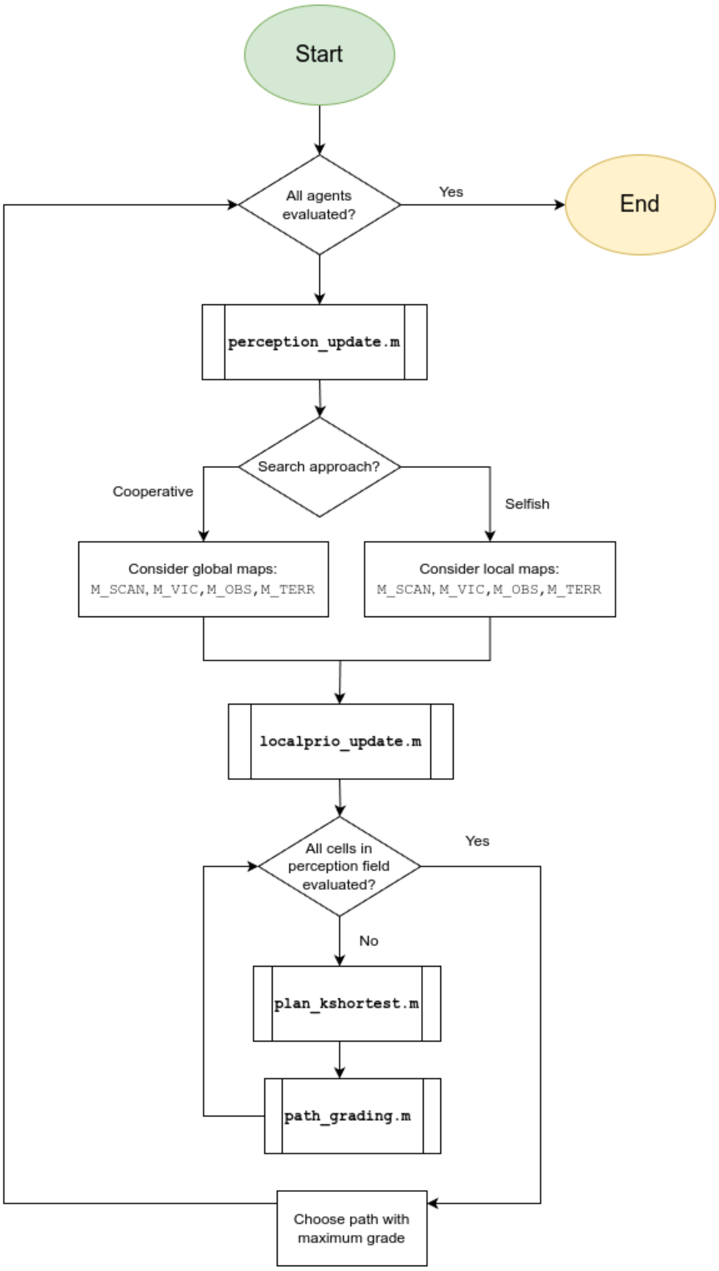

Start
All agents evaluated?
Yes
End
perception_update.m
Search approach?
Cooperative
Selfish
Consider global maps: M_SCAN, M_VIC, M_OBS, M_TERR
Consider local maps: M_SCAN, M_VIC, M_OBS, M_TERR
localprio_update.m
All cells in perception field evaluated?
Yes
No
plan_kshortest.m
path_grading.m
Choose path with maximum grade

Figure A.3: Flowchart for local controller

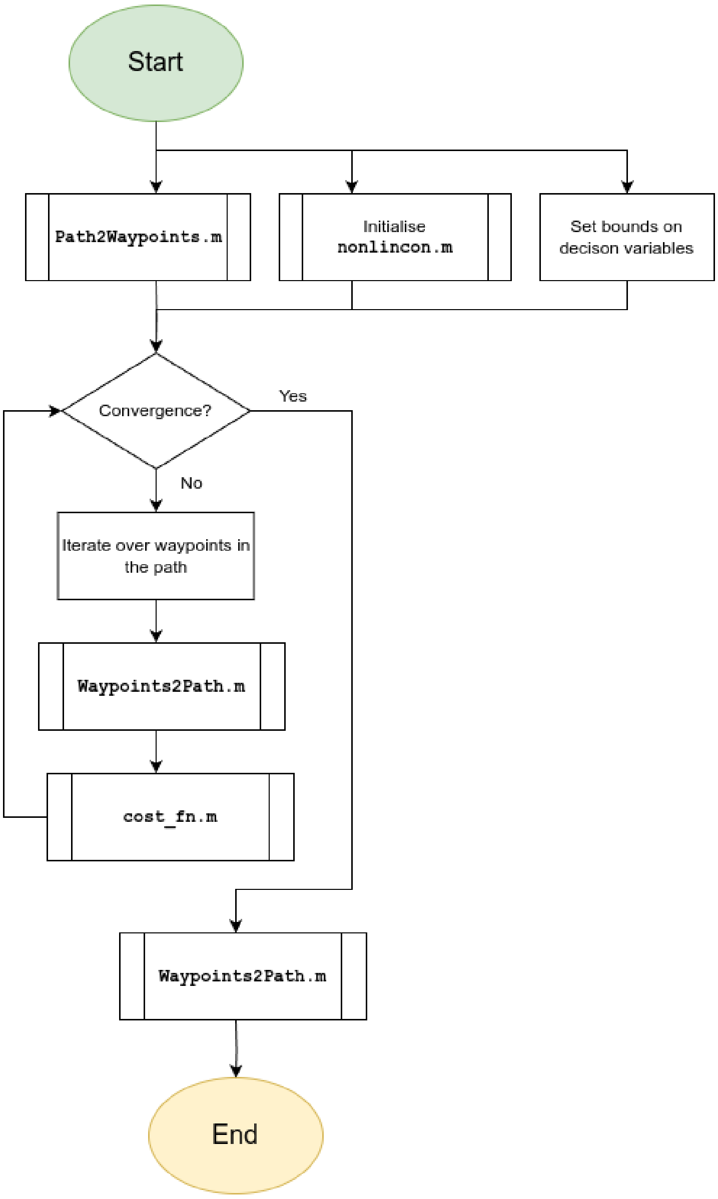


Figure A.4: Flowchart for supervisory controller