Hierarchical Cooperative Mission Planning of Non-Homogeneous Autonomous Search-and-Rescue Robots

A Case Study

Christopher de Koning

TU Delft

Hierarchical Cooperative Mission Planning of Non-Homogeneous Autonomous Search-and-Rescue Robots

A Case Study

by

Christopher de Koning

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday January 20, 2022 at 13:30.

Student number:	4456831	
Project duration:	September 3, 2020 – November 24, 2021	
Thesis committee:	Prof. Dr. Ir. J. M. Hoekstra,	TU Delft, chair
	Dr. A. Jamshidnejad	TU Delft, supervisor
	Dr. O. A. Sharpanskykh,	TU Delft, external examiner

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Preface

With this thesis I conclude more than a year's worth of hard work. Looking back, I can confidently say that I have done my best and that I am proud of the result. As I prepare myself for the somewhat unceremonious event of a fully online MSc defense, I want to give my honest opinion about writing my entire thesis remotely from home. I have generally not enjoyed it as much as the rest of my study. The Covid-19 pandemic has made me realise even more how much my education and personal life has been and still is influenced by the people around me, and my MSc thesis confronted me with this fact daily. Hence, I would be wrong to claim that I have accomplished this milestone all by myself.

First and foremost, Dr. Anahita Jamshidnejad has provided me with invaluable guidance and an incredibly astute eye for detail over the past year. Even when I had doubts about my own work, she was able to point me in the right direction with fresh confidence. I can only thank her for always assuring the academic quality of this research with relentless effort. I would also like to thank my girlfriend Nikki and my roommates Roel and Camille for putting up with my ever-changing mood during the past period. Similarly, my family and friends have never failed to provide a welcome distraction whenever they saw I needed it (whether I agreed with them or not). I hope to repay all their daily love and patience in an appropriate fashion. This holds for my parents in particular, to whom I owe everything I have achieved - I promise to help out with jobs around the house more.

Christopher de Koning
Delft, December 2021

Contents

Nomenclature

ACS	Ant-Colony System
ANN	Artificial Neural Network
ASR	Air-Sea Rescue
CPP	Cooperative Path Planning
DP	Dynamic Programming
FLAPTAPP	Fuzzy-Logic-Assisted Prioritised Task Assignment and Path Planning
FLC	Fuzzy Logic Control
FMRLC	Fuzzy Model Reference Learning Control
HOP	Hindsight Optimisation
LiDAR	Light Detection and Ranging
MILP	Mixed-Integer Linear Programming
MPC	Model-Predictive Control
MRAC	Model-Reference Adaptive Controller
MRTA	Multi-Robot Task Assignment
NMPC	Nonlinear Model-Predictive Control
OLS	Ordinary Least Squares
RPP	Revised Prioritised Planning
SARS	Search-And-Rescue System
SAR	Search-And-Rescue
SLAM	Simultaneous Localisation And Mapping
SMPC	Stochastic Model-Predictive Control
ST-SR-TA	Single-Task, Single-Robot, Time-extended Assignment
TS	Takagi-Sugeno
UAV	Unmanned Areal Vehicle
UMRTA	Uncertain Multi-Robot Task Assignment
UNDRR	United Nations Disaster Risk Reduction
USAR	Urban Search-and-Rescue
WiSAR	Wilderness Search-And-Rescue

I

Research Paper

Hierarchical Cooperative Mission Planning of Non-Homogeneous Autonomous Search-and-Rescue Robots

Christopher de Koning[1*]

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

Corresponding author(s). E-mail(s): C.M.M.deKoning@student.tudelft.nl;

Abstract

The application of autonomous robots in search-and-rescue (SAR) missions forms a challenging field of research. Cooperative search behaviour can greatly increase the efficiency with which a multi-agent system creates situational awareness of and finds victims within an unknown environment. In this research we develop an autonomous mission planning approach that exploits non-homogeneous characteristics of the robots to increase the overall search performance. Furthermore, the proposed approaches incorporate a hierarchical, cooperative control architecture: At the lower level of control, every SAR agent is locally controlled by a heuristic approach that uses fuzzy-logic control (FLC) and A* search to guide its individual actions. At the higher level of control, a model-predictive-control-based (MPC-based) system coordinates the actions of all SAR agents when two or more agents intend on searching the same area at once. When simulated in a virtual SAR environment, we show that the area coverage performance of the cooperative search approach is comparable to a purely heuristic approach designed for area coverage, while simultaneously outperforming this and another optimisation-based approach in victim detection efficiency. This shows that the hierarchical combination of the heuristic local controller and the optimisation-based supervisory controller is able to outperform either one approach individually. Furthermore, it is demonstrated that the cooperative search approach is able to efficiently resolve particular SAR-related search conflicts where a non-cooperative structure fails.

Keywords: Cooperative Search, Hierarchical Control, Search-and-Rescue

1 Introduction

The past couple of decades have seen a rapid increase in the applications of robotics in the field of search-and-rescue (SAR). Recent advances in SAR robots are motivated by a number of advantages of these systems over teams of only human workers. From a safety perspective, the deployment of (autonomous) robots reduces the risks that human rescue workers would otherwise be subject to [1]. Furthermore, human resources can be scarce in an urban SAR scenario, and the automation of search and relief tasks by means of autonomous robots allows for vital human resources to be made available for other tasks such as logistics and victim aid. Finally, fast victim detection time is of the essence in SAR missions [2]. In these disaster settings, fast, agile, and autonomous drones such as unmanned aerial vehicles (UAVs) can be deployed to rapidly map out otherwise inaccessible disaster areas. Within a SAR mission, many different tasks can be performed by robots. One such tasks is gathering

information from an unknown disaster environment to improve situational awareness. Such information may include the location of victims, explosives, or any other elements that may affect the mission. During a SAR mission, situational awareness can prove to be vital for mitigating risks and saving lives [3, 4].

To reduce the time it takes to gather this information, many SAR systems employ a fleet of multiple robots. Initially, these robots were controlled or closely supervised by human operators and thus had limited autonomy. The effective operation of such robots is entirely dependent on the human users developing situational awareness on the robots and environment. Issues such as difficulties in robot localisation, limitations in cognitive resources, and poor integration of data at the human-machine interface lie at the base of human-induced errors in SAR missions [1, 3]. Consequently, autonomy of these multi-agent systems has increased throughout time, which has given rise to a wide variety of SAR-related research [5, 6].

When categorising cooperative multi-agent search approaches, we make a distinction between destination-oriented and coverage-oriented search approaches. The first type commonly assumes the availability of some source of prior knowledge related to the target distribution in the environment. Examples of such search approaches can be found in [7–10]. These are generally destination-oriented search approaches, since they are less concerned with any information the SAR agents are able to gather while heading towards their predetermined targets. These approaches generally perform poorly in an environment where no prior knowledge is available to the SAR agents. In an unknown SAR environment, however, the problem of cooperative search has a lot in common with that of coverage path planning (CPP), since every unknown area is a possible area of interest for the SAR agents. In the past couple of decades, CPP has been addressed in a considerable amount of research. Galceran and Carreras in [11] provide an extensive survey of research in CPP up until 2013. In this field of research, ant-colony system (ACS) algorithms are a popular type of bio-inspired area coverage method, due to their computational efficiency and simple applicability [12, 13]. Other more recent cooperative coverage methods include the use of machine learning and neural networks, in which the system's agents progressively learn better area coverage behaviour [14, 15]. The drawback of such methods is that they require to be trained before they can perform search tasks. Furthermore, it must be noted that most CPP approaches, such as ACS search, do not incorporate victim/target detection in their search behaviour, and are limited to area coverage objectives. Arnold et al. in [16] present a cooperative, multi-agent SAR system that does incorporate both objectives; in addition to directed victim detection, the agents are also able to adopt exploratory behaviour in order to increase situational awareness of the environment. However, the SAR agents are restricted to a set of fixed behaviour types which limits the flexibility of their search strategy.

From a control architecture perspective, another distinction can be made between centralised and decentralised control approaches applied to multi-agent SAR. In centralised methods, the SAR agents communicate their situational awareness to a single control server, which determines the mission plan for all SAR agents. Examples of centralised search approaches can be found in [7–10]. This has the advantage that all computationally demanding tasks can be concentrated in a single ground station, which reduces the cost of the robots used while yielding a higher global mission performance. The disadvantage of centralised approaches, is that they are reliant on a consistent communication link and are vulnerable for single-point failure [17]. Decentralised approaches mitigate these risks by equipping each SAR agent with their own onboard controller, thus making them more robust to loss of communication or failure. However, a distributed approach may yield a sub-optimal solution for the global mission [5]. Examples of decentralised search approaches can be found in [12, 14, 16, 17]. Hybrid or hierarchical control methods can combine the strengths of both centralised and decentralised search approaches. Especially in multi-agent SAR systems, hierarchical control may be beneficial to coordinate the behaviour of multiple distributed (non-optimal) controllers, while being robust against single-point

failure. Little research is available of such hierarchical approaches in search (and rescue) applications. Examples of such hierarchical control architectures include [18–20], however these approaches are limited to destination-oriented target search and task allocation.

This research focuses on the multi-agent, cooperative search problem performed by a fleet of non-homogeneous UAVs. More specifically, this research is interested in developing a hierarchical mission planning approach that exploits the non-homogeneous sensory capabilities of each agent to increase the search performance of the SAR system. The mission task consists of maximising situational awareness of an environment, by maximising both area coverage and victim detection efficiency in the least amount of time. The aim of this research is to formulate a novel type of control approach to achieve this.

The remainder of this paper is structured as the following: the contributions of this research are highlighted in Section 2, which is followed by defining the search problem and its corresponding terminology in Section 3. Next, Section 4 discusses the proposed ideas for developing a hierarchical mission planning controller for SAR robotics. This is followed by Section 5, which describes the case study and experimental setup that have been used to evaluate the performance of the proposed controller. After presenting the results of these experiments, we analyse and discuss them in Section 6. Finally, we summarise the main contents of this paper and propose recommendations for future research in Section 7.

2 Main Contributions

The main contributions of this research to the state-of-the-art are twofold:

1. We present a new control framework for multi-agent SAR, that utilises non-homogeneous sensor characteristics of SAR agents in its search strategy. Furthermore, this search strategy includes both directed victim search and area coverage tasks.
2. This research formulates a novel hierarchical control approach, which combines distributed local controllers for all SAR agents with a centralised supervisory controller to coordinate the global mission. More specifically, the local controllers use an FLC-based heuristic, whereas the supervisory controller uses an MPC-based approach.

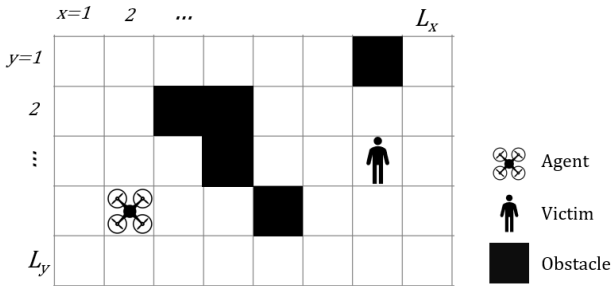

Fig. 1: Simulated SAR environment.

3 Problem Formulation

In this section, we define and formulate the mission planning problem for cooperative non-homogeneous SAR agents. More specifically, we discuss the modelling of the SAR environment in Section 3.1. This is followed by a description of the SAR agents and their sensory capabilities in Section 3.2 and Section 3.3, respectively. Finally, Section 3.4 covers the modelling of the victims present in the SAR environment.

3.1 SAR Environment

The SAR environment E is modelled as a 2D bounded cellular area of $L_x \times L_y$ cells. More specifically, a cell corresponds to the coordinates (x, y) of its centre. Each cell (x, y) is defined by a number of quantified properties, including the *scan certainty* value for time step τ, denoted by $c(x, y, \tau) \in [0, 1]$. The scan certainty value is determined based on the degree of information that is known of the contents of that cell. This depends on whether or not the cell has ever been visited and scanned by an agent and if so, how accurate the sensed or measured data is estimated to be. If $c(x, y, \tau) = 0$, the cell is completely unknown for a SAR agent, whereas $c(x, y, \tau) = 1$ implies that an agent has full knowledge of that cell. The scan status of each cell in the environment is included in a matrix called the *scan certainty map* $\mathcal{C}$. Dependent on the control approach guiding the SAR agents (elaborated upon in Section 4), $\mathcal{C}$ is either a globally available map or a local map of information acquired and used by each SAR agent.

Each cell can either be empty, or be occupied by static obstacles (i.e., walls, pillars, rubble), dynamic victims, or SAR agents. We assume that for any time step, a cell can be resided by a single victim only. Furthermore, static obstacles make an area inaccessible for an agent or a victim to occupy or pass through. If a cell is occupied by such an obstacle, it will be included in a map called the *occupancy map* $\mathcal{W}$. A schematic representation of the SAR environment is shown in Figure 1.

3.2 SAR Agents

For this research, we consider a multi-agent SAR system composed of N non-homogeneous, rotary UAVs, denoted by $\mathrm{a}_i = \mathrm{a}_1, \mathrm{a}_2 \ldots \mathrm{a}_N$. The agents may be different from each other in two properties regarding their sensory capabilities. Firstly, SAR agents may have different perception radii $r_{\mathrm{p},i}$. Note that those cells of the environment E that fall within the circle of radius $r_{\mathrm{p},i}$, that is centred at the current position of the SAR agent, form the *perception field* of agent i (see Figure 2). The perception field of a SAR agent is denoted by E_{a_i}, where $E_{\mathrm{a}_i} \subseteq E$. Secondly, the *perceptual uncertainty reduction rate* η_i for various SAR agents may be different, where this parameter is an indication of the accuracy of an agent's sensor. The perceptual uncertainty reduction rate will be discussed in more detail in Section 3.3.

For every time step, each SAR agent is described by its position in the grid environment E (which corresponds to the cell the SAR agent is flying above) and can move to one of its 8 neighbouring cells through one of the 8 directions (north, north-east, east, south-east, south, south-west, west, north-west) shown in Figure 2.

During the SAR mission, the agents are assumed to have a sufficient degree of manoeuvrability, such that they have a turn radius that is smaller than the cell they occupy, and that they can remain in a stationary (hovering) position in a cell if needed. Hence, it assumed that an agent can at any time move in any of the 8 possible directions, regardless of its orientation. Furthermore, it is assumed that the agents can avoid colliding with each other by adjusting their altitude. This way, two agents can (briefly) pass through the same cell in the 2D plane.

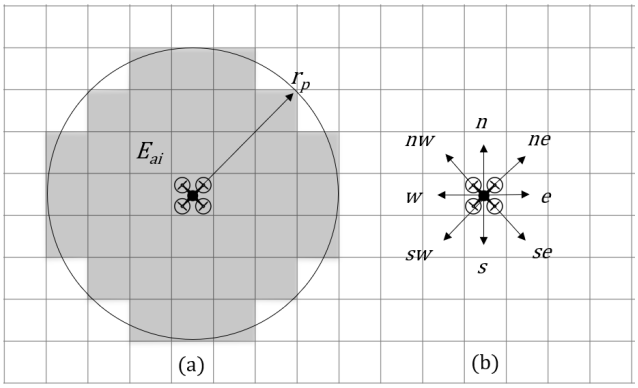

Fig. 2: Perception field of an agent (a) and its movement possibilities (b).

3.3 Sensory Capabilities of SAR Agents

All agents have a sensory base consisting of two main sensor types: First, each agent is equipped with (optical) cameras that can be used to detect victims and to gather visual information of the environment. This information is stored in the global scan certainty map $\mathcal{C}$. Secondly, SAR agents are equipped with sensors for detecting WiFi-enabled devices. Studies such as [21] are focused on locating disaster victims by means of the wireless signal of their passive mobile phones. For this research, we assume that the SAR agents use their WiFi sensors for both victim localisation and assessment of a victim's health condition. However, these requirements can also be met with alternative sensors, such as acoustic, heat, and optical sensors, or a combination thereof [1, 22].

The sensors of each SAR agent may in general be imperfect, meaning that a single scan may not yield full certainty/knowledge of all cells in the perception field of the agent. We consider two sources of sensory imperfections: (i) One visit and complete round of scanning of a particular cell for a single time step may not provide full certainty due to structural imperfections in the agent's sensors. In this case, following Dempster's rule of combination [15], for every time step the *uncertainty* of a cell is reduced by a perceptual uncertainty reduction rate, which we show by $\eta_i \in (0, 1]$ for SAR agent i. (ii) While all cells within the perception field E_{a_i} of SAR agent i will be scanned by the agent's (possibly imperfect) sensor, the rate change of scan certainty also decreases

with increased distance from the SAR agent/sensor. Therefore, the *proximal uncertainty reduction rate* $\sigma(x, y, \tau)$ for a cell (x, y) for time step τ is a function of the Euclidean distance $r_i(x, y, \tau)$ of the cell from the SAR agent i for time step τ. In other words, while the perceptual uncertainty reduction rate depends on which SAR agent is reporting the value (i.e., it is agent/sensor-related), the proximal uncertainty reduction rate is a distance-related property. Mathematically speaking, we can write:

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

with

$$\sigma_i(x, y, \tau) = 1 - e^{-r_i(x,y,\tau)}\left(1 - \eta_i\right) \cdot \frac{1 - \text{sign}\left(r_i(x, y, \tau) - r_{\text{p},i}\right)}{2} \tag{2}$$

where we suppose that the proximity effect on the rate of the scan certainty follows an exponential decay function. In (2), $\sigma_i(x, y, \tau)$ is the share of the proximal uncertainty reduction rate of cell (x, y) due to the information that is provided via the sensor of SAR agent i. Based on the proposed formulation (2), when $r_i(x, y, \tau) = 0$, i.e., for the cell that SAR agent i is currently located in, the proximal uncertainty reduction rate corresponding to agent i is equal to η_i. Moreover, when $r_i(x, y, \tau) \geq r_{\text{p},i}$, based on (2) the proximal uncertainty reduction rate corresponding to agent i is 1. Note that in (1) the overall effect of all the N SAR agents that are in the SAR environment is considered. To prevent a 0 proximal uncertainty reduction rate - corresponding to a SAR agent that is too far from cell (x, y) to be able to scan this cell - to falsely affect $\sigma(x, y, \tau)$, we have used the max function in (1).

Finally, the updated degree of certainty regarding cell (x, y) can be computed via:

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

with

$$z(x, y, \tau + 1) = \sigma(x, y, \tau) z(x, y, \tau) \quad \forall (x, y) \in E_{\text{a}_i} \tag{4}$$

which incorporates both sources of sensory imperfection that have been explained above. Here, the degree of uncertainty of a cell $z(x, y, \tau)$ forms the complement of $c(x, y, \tau)$.

3.4 Victim Modelling

The simulated environment contains a fixed and unknown number of V victims with location and health condition that are also unknown to all agents. When simulating victims in this research, two main characteristics are modelled. First, random victim movement is considered. This victim behaviour is added to the simulation in order to evaluate the robustness of the mission planning module when subjected to dynamic uncertainty. For every time step, a victim moves to a neighbouring cell with a probability of $8 \times P_{\text{vm}}$. Thus, the movement of a victim is composed of 8 individual, equal movement possibilities to either one of that victim's neighbouring cells.

Secondly, each victim has a *victim health state* $h_V(\tau) \in [0, 100]$, which corresponds with how healthy or injured a victim is at time step τ. When a victim is detected by a SAR agent, their initial health state is registered. Over time, h_V may decrease, which simulates the declining health condition of victims when they remain unaided. The rate of the changes $\dot{h}_V$ in a victim's health state is assumed to follow the following relation:

$$\dot{h}_V(\tau) = \begin{cases} -0.25 & 30 \leq h_V(\tau) \leq 100 \\ \frac{h_V(\tau)}{60} - 1 & 0 \leq h_V(\tau) \leq 30 \end{cases} \tag{5}$$

Based on (5), a victim has a uniformly deteriorating health condition for $30 \leq h_V(\tau) \leq 100$. However, whenever $h_V(\tau) \leq 30$, the rate of change of the victim's health state decreases linearly until the victim perishes at $h_V = 0$. This simulates the accelerated deterioration of victims with a more critical health condition, whereas victims with a more stable condition can survive unaided for a longer period of time. The values of the victim's health state degradation rate are chosen in such a way, that an exhaustive search strategy of the experimental environment described in Section 5 cannot guarantee the survival of all victims. Hence, this introduces a time constraint to the SAR mission.

In this research, a SAR agent detects a victim only when they are both present in the same cell. When a victim is detected, the SAR agent stores the location, health state, and time of detection in its own matrix called the victim map $\mathcal{V}_i$. By locally storing this information, each SAR agent only has knowledge about the victims that it has detected itself.

4 Control System

This section explains the architecture of the proposed hierarchical control system that guides the search behaviour of the SAR agents, as well as the control methods that are used in the various control levels. The controller has two levels of hierarchy, as shown in Figure 3. The distributed lower-level controllers dictate the local search behaviour of each SAR agent (see Section 4.1 for details), while the centralised supervisory controller coordinates the behaviour of the entire fleet of SAR agents, such that search conflicts between the SAR agents are resolved (see Section 4.2 for details). In this diagram it is also shown that the SAR agents only communicate with the supervisory controller, and do not share information among themselves. This architecture combines the strength of both centralised and distributed control approaches; each local controller will be able to independently guide its corresponding SAR agent, while the supervisory controller coordinates the global mission of all SARS agents when required. This increases the robustness of the SAR system against (communication) failure, while containing heavy processing tasks in a centralised control module that considers the global mission of the entire system.

4.1 Local Controller

The local controller of each agent consists of two main parts. First, a SAR agent processes its own local sensory data to form a priority map of its perception field, which indicates how valuable each cell is for the SAR agent to visit. Next, the controller chooses the path that yields the highest local gain in this priority, which is expressed by means of a path grade. The grade that each path receives, represents how favourable a path is. For this, we must recall that the mission objective is to

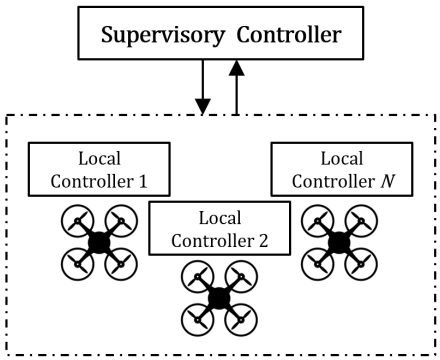

Fig. 3: Hierarchical architecture of the cooperative mission planning controller.

maximise area coverage and victim detection efficiency in the least amount of time. From this, we consider two main criteria when selecting a path:

1. **Reduction of time**: the SAR agent wants to reach targets of interest in the shortest possible time in order to minimise the overall SAR mission time.
2. **Increase of exploration**: the SAR agent wants to scan as many other (unexplored) cells along the way as possible, in order to maximise area coverage.

The two objectives mentioned above may possess a conflict: for the first objective, the SAR agent is destination-oriented and aims to reach a specific cell in the shortest possible time, while for the second objective the SAR agent is coverage-oriented, since every cell that the SAR agent visits on the path to its destination is worth being visited. The local controller is designed on the premise of meeting both objectives.

4.1.1 Search Priority Assignment

The first main module within the local controller of each agent is that of task priority assignment. In the context of this research, *scanning a search area* is the only type of SAR task considered. From this, it follows that task priority is defined as the urgency to scan a given cell within the environment.

For each cell in E_{a_i} a priority score is determined using fuzzy-logic control (FLC). The motivation for this control approach, is that FLC is an efficient heuristic control method that is able to mimic human decision logic and express it in

mathematical terms. Due to the limited computational power available on board of the SAR agents, it is necessary to choose a computationally efficient control approach. Furthermore, FLC was chosen for the local SAR agent controllers due to the strong link between SAR mission planning and human operators. Currently, human rescue workers are still the primary controllers when it comes to allocating resources in a SAR mission. Hence, it is argued that an effective controller tries to mimic this human way of thinking, with the advantage that the knowledge of human SAR personnel can be directly incorporated in the control approach. To achieve this, a rule-based Mamdani fuzzy inference system (FIS) is used for the local controller of the SAR agents. The set of M if-then rules that make up the rule base has the following structure:

$$\begin{aligned}\mathcal{R}_m: \quad & \textbf{If } P_V(x,y) \text{ is } A_{m,1} \textbf{ and } h_V(\tau) \text{ is } A_{m,2} \\ & \textbf{and } c(x,y,\tau) \text{ is } A_{m,3} \textbf{ then } \rho(x,y) \text{ is } B_m \\ & m = 1, 2, \dots M, \quad (x,y) \in E_{a_i} \end{aligned} \tag{6}$$

In this type of fuzzy model, both the antecedent propositions A_m and consequent propositions B_m are fuzzy sets, which are mathematically formulated by their corresponding membership functions. In the task priority assignment module, the fuzzy system takes 3 inputs. For these inputs, it is important to note that each SAR agent only has its own local knowledge available, contained in its scan certainty map $\mathcal{C}_i$ and victim map $\mathcal{V}_i$. Consequently, one SAR agent does not know which victims are detected or which areas have already been scanned by another agent.

The first input, the victim health state h_V, has been discussed in Section 3.4 and has fuzzy membership functions *Critical, Medium,* and *Stable*. The scan status c has been discussed in Section 3.1 and Section 3.3 and has fuzzy membership functions *Unknown, Partial,* and *Known*. Next, the victim probability P_V is defined as the probability that the contents of a particular cell are correctly identified as a victim. This differs from the scan certainty, which indicates how completely the general contents of a cell have been scanned. The victim probability, however, relates to specifically searching for victims, as opposed to the overall area coverage of the environment. Similarly to the scan certainty, we assume that P_V is prone to proximity-related sensor inaccuracies and is thus a function of the distance between an agent and a possible victim. It is supposed that P_V follows a basic quadratic loss function:

$$P_V(r) = -\left(\frac{r}{r_p}\right)^2 + 1 \qquad r \in [0, r_p] \tag{7}$$

Cells that are not expected to contain a victim receive a value of $P_V = 0$. The victim probability has membership functions *Low, Medium,* and *High*.

The output of the FIS is the search priority score ρ for each cell in E_{a_i}, which represents the level of interest/urgency to scan that cell. The priority score has fuzzy membership functions *Very Low, Low, Medium, High,* and *Very High*. All membership functions are shown in Figure 4. Furthermore, following the structure in (6), the complete rule base in ascending order of priority is shown in Table 1. In general, cells are given a higher priority if they are likely to contain a victim and if that victim is expected to be more hurt. Similarly, cells that have not yet (extensively) been scanned also receive a higher priority score. Finally, cells contained in $\mathcal{W}$ (thus being inaccessible to SAR agents) are given a priority of 0.

4.1.2 Path Planning

The second part of the local mission planning controller, is the path planning module. Since time is of the essence in the SAR mission and the SAR agent is thus partially destination-oriented, the local controller takes a shortest-path approach as a basis for its path planning module. However, since the SAR agent is also partially coverage-oriented, at this stage every cell within its perception field is still considered a potential target.

The first part of the path planning module is to determine the globally shortest path p_i^1 from the SAR agent's position s_i to any given target cell t_i within the perception field of SAR agent i. In this research, the path planning module makes use of an A* Search approach [23], which can be seen as an extension of Dijkstra's algorithm [24]. Contrary to the algorithm it is based on, A* Search limits the number of path possibilities by avoiding expanding paths that are already costly when

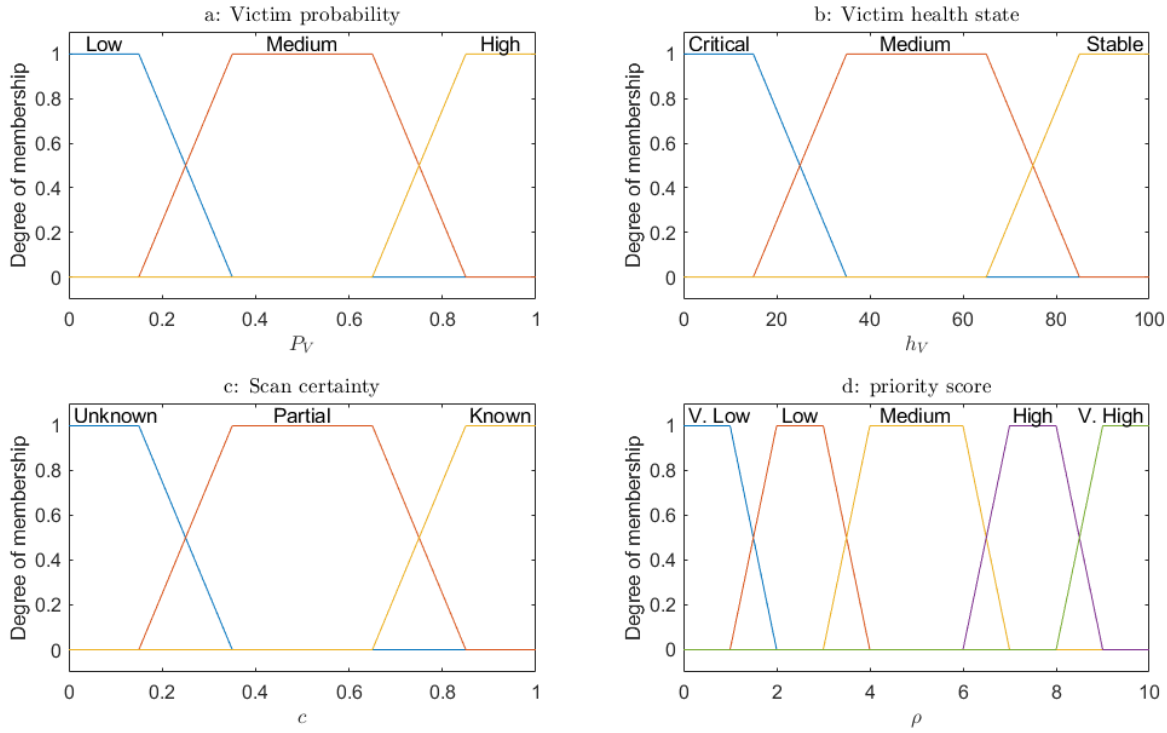

Fig. 4: Fuzzy membership functions of the FIS inputs and output.

Table 1: FIS rule base

$\mathcal{R}_i$	P_V	h_V	c	ρ
1	Low	Stable	Known	Very Low
2	Low	Medium	Known	Very Low
3	Low	Stable	Partial	Very Low
4	Low	Medium	Partial	Low
5	Low	Critical	Known	Low
6	Medium	Medium	Partial	Low
7	Medium	Critical	Known	Low
8	Low	Stable	Unknown	Low
9	Medium	Stable	Known	Medium
10	Medium	Medium	Known	Medium
11	Medium	Stable	Partial	Medium
12	High	Stable	Partial	Medium
13	High	Medium	Known	Medium
14	High	Critical	Known	Medium
15	Low	Critical	Partial	Medium
16	Low	Medium	Unknown	Medium
17	High	Stable	Known	High
18	Medium	Stable	Unknown	High
19	Medium	Medium	Unknown	High
20	High	Stable	Unknown	High
21	Low	Critical	Unknown	High
22	Medium	Critical	Partial	Very High
23	High	Medium	Partial	Very High
24	Medium	Critical	Unknown	Very High
25	High	Medium	Unknown	Very High
26	High	Critical	Partial	Very High
27	High	Critical	Unknown	Very High

extended to the end goal. The algorithm does this by including a heuristic when evaluating each path possibility, which it then uses to only expand paths that move roughly towards the goal point, and neglect paths that only move away from the goal. This greatly reduces the computational effort and time required to find the optimal path, especially when it is applied to equally distanced grid maps.

Once the shortest path between the agent and target cell has been determined, it is possible to find the next shortest alternatives to this path. This is motivated by the coverage-oriented part of the SAR agent: by determining a set of K permutations on the shortest path, the SAR agent is offered more possible paths that are comparable in length but may pass through different areas of interest. This is done with Yen's algorithm [25], as it ensures an efficient method for determining the K guaranteed shortest paths $p_i^k = \{p_i^1, p_1^2, \dots p_i^K\}$ for agent i. The algorithm does this by systematically closing subsequent nodes within the previously found shortest path p_i^{k-1}, and evaluating the path permutations this results in. It stores these permutations in a set, and chooses the shortest path in this set as the next shortest path p_i^k. The algorithm repeats this until path

$k = K$ has been determined. All shortest paths are determined while avoiding obstacles in the environment, by omitting cells contained in $\mathcal{W}$.

4.1.3 Path Grading

Once the K shortest paths have been determined for each possible target cell, it is necessary to grade each possible target-path combination, and determine the best action for the agent to execute. As mentioned previously, the grade that each path receives represents how favourable a path is. This grade is based on the level of *exploration* and the *time duration* of a path. These elements form conflicting objectives within the SAR mission, and will both compete in the path grade in some form. For instance, a very short path may lead an agent through an area that has a low task priority, or a path that visits many high-priority cells may take a lot of time to complete.

Each of the aforementioned pillars must be expressed in mathematical terms, in order to effectively grade a path. *Time* is a relatively straightforward aspect, since the travel time of the agent can be derived from both the path length and the agent's speed. Since it is assumed that an agent can at most move one cell at every time step, the path duration L_p is simply equal to the Euclidean distance of a path.

Secondly, a path is also graded based on its area coverage, or exploration. The most straightforward method would be simply to count the number of cells that the agent visits along the way, however this could encourage the agent to visit many low-priority cells before finally reaching the goal cell which may have the main priority. There is a risk that such behaviour defeats the purpose of prioritising the end goal. To prevent this, exploration is represented by defining the *discounted return* R_{p_i} of a path, as shown in (8):

$$R_{p_i} = \sum_{\tau=0}^{H_{p_i}} \gamma^{\tau} \rho_{\tau} \tag{8}$$

The discounted return is the sum of the rewards that an agent receives from its action sequence. In this application, the actions of an agent i are the cells that it visits during its prediction horizon H_{p_i}, and the rewards ρ_{τ} are the priority scores of each cell. Rewards that are further in the future are discounted with a discount factor γ^{τ}, with $\gamma \in [0, 1]$. This can be interpreted as holding less value to more uncertain future actions, while at the same time giving a higher grade to paths that visit higher priority cells sooner. Exploration is still encouraged, since visiting more cells will yield a higher return, however it now also stimulates the preference to visit mostly high-priority cells as soon as possible.

With all inputs defined, it is possible to formulate a function to evaluate the path grade G_{p_i} of each agent. For the initial grading function, a weighted summation of the inputs is chosen:

$$G_{p_i} = -\alpha L_{p_i} + \beta R_{p_i} \tag{9}$$

In this relation, the weights α and β are constant values. Furthermore, it can be observed that R_{p_i} acts as a stimulus for the path grade, since more rewarding paths (in terms of search priority) are preferred. However, L_{p_i} imposes a penalty on the grade, since a longer travel time corresponds to a less favourable path.

4.2 Supervisory MPC-Based Controller

To perform the cooperative SAR mission, the agents are guided by an additional supervisory controller. It is assumed that an external, centralised server performs the supervisory control task, while it both receives information from and sends out control outputs to the SAR agents. As mentioned previously, the supervisory controller is mainly tasked with resolving search conflicts between SAR agents. By definition, in this research, a search conflict occurs when the perception fields of two or more SAR agents have an intersecting area above a certain threshold, denoted by $E_{\mathrm{a}_i} \cap E_{\mathrm{a}_j} \geq I$. Only when this occurs, the supervisory controller is activated and coordinates the global mission for all SAR agents.

In the hierarchical architecture of the developed controller, a model-predictive control (MPC) approach is considered for the supervisory controller. A process model of the SAR environment is used to predict the system output of a sequence of actions over a prediction horizon H_p. By optimising a given cost function, the supervisory controller obtains an optimal action sequence for the SAR agents to conduct. This optimal sequence is the set of optimal paths for all SAR agents to follow. The controller then

implements the first action of this sequence, corresponding to the optimal next step for all SAR agents. On a mathematical level, the supervisory controller is thus optimisation-based, as opposed to the (sub-optimal) heuristic controllers of the individual SAR agents. The rationale is that the local heuristic controllers can, in most cases, efficiently find a satisfactory solution to guide each SAR agent individually at a given time step. In the less frequent case of a search conflict, it is presumed that the local controllers alone are unable to efficiently spread the SAR agents over the environment. Hence, the optimisation-based supervisory controller is activated to find the optimal joint pay-off for all SAR agents, at the cost of being more computationally demanding. However, since the supervisory controller is contained on an external server, it is argued that it is possible to utilise a more computationally demanding control approach than for the local controllers of the SAR agents. Additionally, the supervisory controller can then use the set of paths determined by each SAR agent locally as an initial 'warm start' to the optimisation problem. The reason for this is to help the supervisory controller converge faster to an optimal solution, thus saving time on the optimisation computations while making extra use of computations made by the local controllers.

4.2.1 MPC Objective Function

The proposed objective function to be maximised for the supervisory controller is given by:

$$J(P) = w_1 \sum_{i=1}^{N} G_{p_i} + w_2 \sum_{(x,y)\in E} c(x, y, H_p) \quad (10)$$

where P is the set of paths of all N SAR agent, and forms the design variable for the MPC optimisation. It can be seen that the objective function is a weighted sum of two terms: (i) the overall grade of all individual paths (estimated by (9)) and (ii) the total predicted scan certainty of the entire environment at the end of the prediction horizon H_p. By incorporating this term in the objective function, the fleet of agents is encouraged to spread out over the environment, since this yields a larger predicted increase in summed scan certainty. This enforces that the supervisory controller maximises the combination of locally beneficial paths for each agent, as well as a globally beneficial fleet distribution for area coverage and determine a set of paths for the SAR agents that results in a balanced trade-off between these locally and globally optimal solutions.

Contrary to the local controllers, the supervisory controller uses the merged knowledge of all SAR agents in its optimisation strategy, by means of its communication link between all SAR agents and the supervisory controller (shown in Figure 3). As a result, the supervisory controller is able to log and use the global scan certainty map $\mathcal{C}$ of the environment, as well as merge all local victim maps into a global victim map $\mathcal{V}$ containing the information of all previously detected victims.

4.2.2 Optimisation Problem Formulation

The optimal control problem, corresponding to an MPC formulation can be defined as follows:

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

$$\textbf{s.t.} \quad P = \{p_i, \quad i = 1, \dots N| \quad p \in \mathcal{P}\} \quad (11a)$$

$$p_i = \{(x^{\mathrm{a}}_{1,i}, y^{\mathrm{a}}_{1,i}), (x^{\mathrm{a}}_{2,i}, y^{\mathrm{a}}_{2,i}), \dots (x^{\mathrm{a}}_{n,i}, y^{\mathrm{a}}_{n,i})| \quad (x, y) \in E, \quad (x, y) \notin \mathcal{W}\} \quad (11b)$$

$$(x^v_j, y^v_j) \notin \{p_Q \cap p_R| \quad Q, R \in \{1, \dots N\} \quad \wedge Q \neq R\} \quad \forall \quad j = 1, \dots V \quad (11c)$$

$$(x^{\mathrm{a}}_{1,i}, y^{\mathrm{a}}_{1,i}) = s_i \qquad \forall \quad i = 1, \dots N \quad (11d)$$

In which $J(P)$ is given by (10) and constraint (11a) states that the paths of all N SAR agents should be contained in $\mathcal{P}$, which denotes the set of all feasible paths. In other words, each path p_i must consist of n adjacent and consecutive cells. Furthermore, constraint (11b) defines that each cell in the path should be within the system bounds of the environment E and cannot be occupied by an obstacle contained in $\mathcal{W}$. Next, constraint (11c) restricts multiple agents from visiting the same victim, by stating that the location of every victim (denoted with superscript v) can only be present in the path of at most one SAR agent. This aims to further improve victim search efficiency and area coverage of the fleet. Finally, constraint (11d) fixes the initial starting cell of each path during the optimisation. This starting cell coincides with the SAR agent's current location, denoted as s_i.

5 Case Study

This section describes the set of numerical simulations designed to evaluate the performance of the proposed control approach. First, Section 5.1 discusses and motivates a number of alternative search approaches used as comparison for the proposed search approach. Next, these approaches are evaluated in both randomly initiated simulation environments in Section 5.2, as well as a few specially designed simulation cases in Section 5.3. These tailored cases simulate specific SAR situations for which the proposed search approach is designed, in order to showcase certain search behaviour of this controller. Furthermore, the computational time of each approach is evaluated. All the experiments are conducted in MATLAB R2019b on a PC with Intel Core i7 Processor with 2.20GHz frequency.

5.1 Comparison Search Approaches

The hierarchical control structure that includes both the local SAR agent controllers and the supervisory controller is termed the *cooperative* search approach, since the inclusion of the supervisory controller allows for a degree of coordination between the SAR agents. Additionally, we compare the cooperative case with four other search approaches. We hypothesise that the combination of the local heuristic controller and the supervisory MPC-based controller performs better than either approach separately, in terms of victim detection, area coverage, and computational effort. Hence, the comparative search approaches aim to validate this theory.

First, an identical set of experiments will be conducted excluding the supervisory controller. This search approach is termed the *selfish* search approach, since all agents are guided only by their own local controller. The selfish approach serves primarily as a reference base to determine the added effect of the supervisory control module on the performance of the SAR system.

The remaining three approaches are a purely optimisation-based, a purely heuristic, and a random search approach. For the purely optimisation-based search approach, the MPC structure of the supervisory controller as described in Section 4.2 is taken as a basis. Contrary to the cooperative search approach, this pure-MPC method solves the optimal control problem for each time step and does not receive the solution from the local controller as a warm start. Instead, the pure-MPC method randomly initialises P_0. For the heuristic approach, an ant colony system (ACS) search method is used, formulated using the basic principles from Koenig and Liu in [12]. For the pheromone map of the agents, the global scan certainty map $\mathcal{C}$ is used to indicate to what extent certain areas have been visited/scanned. Hence, the ACS will also represent a type of (passive) cooperative search approach, since all agents will share information relating to the global mission. Finally, the fourth comparison approach uses a random search strategy for each agent. Although random search can be viable in an environment that is initially completely unknown, this simple search strategy serves primarily as a reference base for the other search methods.

5.2 Random Environment Simulations

The cooperative search approach is tested in a set of 20 simulation cases with a run-time of τ_{max} and a seeded random placement of both victims and obstacles in an environment of fixed size. All environment characteristics are shown in Table 2. Furthermore, this table contains the values for all parameters related to victim and sensory modelling, as well as the hyper-parameter values used in the control approach. For the experiments conducted in this research, we consider a number of $N = 2$ agents. As mentioned in Section 3, these agents are non-homogeneous in their sensory perception radius r_{p_i} and perceptual uncertainty reduction rate η_i. Additionally, both agents start at fixed coordinates for each simulation case. The values of these parameters are shown in Table 3. At the start of each simulation, the agents have no a priori information about the environment, i.e. $c(x, y, \tau_0) = 0 \quad \forall (x, y) \in E$.

As mentioned in Section 4, the controller is designed for both area coverage and victim search efficiency, which by nature form conflicting objectives for the agent. To evaluate the search approaches, we present a number of performance indicators, which show the effectiveness of the search approaches in terms of both area coverage, victim search efficiency, and computational efficiency. Area coverage is evaluated by means of two

Table 2: Environment and modelling parameters

Parameter	Value	
(L_x, L_y)	(40,25)	
V	25	
P_{vm}	5.0	%
K	5	
γ	0.6	
α	2.0	
β	5.0	
I	30	
w_1	1.0	
w_2	0.050	
τ_{max}	300	

Table 3: Agent parameters for all different simulation types.

Random	r_p	η	$(x,y)_{\tau_0}$
$\mathbf{a_1}$	6	0.1	(1,16)
$\mathbf{a_2}$	4	0.3	(13,25)

Case 3	r_p	η	$(x,y)_{\tau_0}$
$\mathbf{a_1}$	7	0.1	(9,10)
$\mathbf{a_2}$	3	0.3	(5,8)

Case 1	r_p	η	$(x,y)_{\tau_0}$
$\mathbf{a_1}$	6	0.1	(10,8)
$\mathbf{a_2}$	4	0.3	(10,6)

Case 4	r_p	η	$(x,y)_{\tau_0}$
$\mathbf{a_1}$	4	0.1	(8,7)
$\mathbf{a_2}$	4	0.3	(8,9)

Case 2	r_p	η	$(x,y)_{\tau_0}$
$\mathbf{a_1}$	6	0.1	(6,10)
$\mathbf{a_2}$	4	0.3	(6,8)

Combi Case	r_p	η	$(x,y)_{\tau_0}$
$\mathbf{a_1}$	7	0.1	(9,17)
$\mathbf{a_2}$	3	0.3	(5,15)

performance indicators. First we consider the total scan certainty of the environment as function of time, defined as:

$$S(\tau) = \sum_{(x,y)\in E} c(x,y,\tau) \tag{12}$$

which is shown in Figure 5 for all approaches. Secondly the rise time of total scan certainty. In Table 4, it is shown how many time steps are required to reach a total scan certainty of $50\%, 70\%, 80\%, 85\%$, and 90% for each search approach. These performance metrics show both the absolute area coverage performance of each search approach, as well as how fast (and thus efficiently) each approach is able to achieve this.

In terms of victim search efficiency, the performance of each search approach is evaluated by means of (i) the number of found and deceased victims at the end of each simulation, (ii) the time step at which each victim is detected, and (iii) the health state of each victim at time of detection. The results of all these performance indicators are shown in Figure 7.

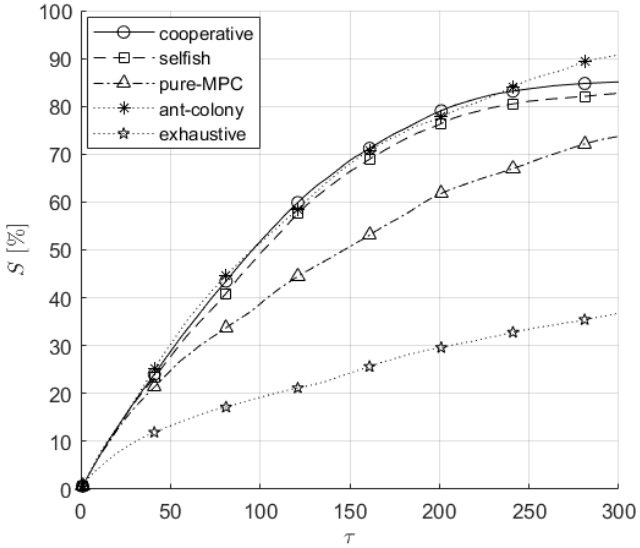

Fig. 5: The total scan certainty for each search approach in a random environment; S is given as a percentage of the maximum achievable scan certainty of the entire environment.

Finally, Table 5 shows the average time required to run a complete simulation of τ_{max} time steps for each search approach. Additionally, Figure 6 shows the number of times a search conflict has been registered by the SAR system using both the cooperative and selfish search approach. For the cooperative approach, this value is coincidentally the number of times the supervisory controller has been activated. In combination with the previously mentioned performance indicators, these results show the cost of area coverage and victim detection performance as function of processing effort.

5.3 Special Simulation Cases

In addition to the random environment simulations discussed previously, we apply the cooperative and selfish search approach to a number of specific simulation scenarios. These small-scale simulation cases place a non-homogeneous, multi-agent system in different situations with conflicts in local versus global mission control. The aim of these special simulations, is to validate that the developed search approach shows the search behaviour that it was designed for. These cases

Table 4: Time steps needed to reach various degrees of total environment certainty in a random environment.

	$\tau_{S=50\%}$	$\tau_{S=70\%}$	$\tau_{S=80\%}$	$\tau_{S=85\%}$	$\tau_{S=90\%}$
Cooperative	97	157	208	291	-
Selfish	104	167	238	-	-
ACS	97	159	218	249	292
Pure-MPC	143	251	-	-	-
Random	-	-	-	-	-

Table 5: Average computational time for a single simulation of $\tau_{max} = 300$ time steps.

Cooperative	**Selfish**	**ACS**	**Pure-MPC**	**Random**
22min 26s	15min 58s	0min 2s	57min 11s	0min 2s

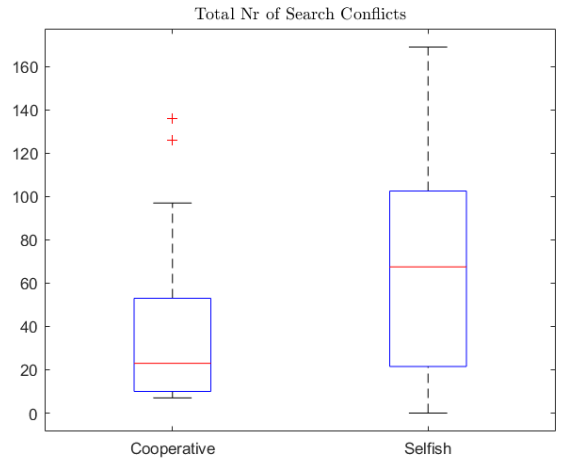

Fig. 6: Number of registered search conflicts using both the cooperative and selfish search approach.

have the same modelling parameters as specified in Table 2, unless specified otherwise. Furthermore, Table 3 lists all relevant agent parameters for each simulation case.

5.3.1 Case 1: Victim-Victim Conflict

In the first special simulation case, 2 agents are tasked with finding 2 victims in an environment of $(L_x, L_y) = (15, 15)$ which is partially known, in which $c(x, y, \tau_0) = 0.20 \quad \forall x, y \in E$. In this setting, victim V_1 has a health state of $h_1(\tau_0) = 10$, and victim V_2 has a health state of $h_2(\tau_0) = 50$. This simulation case is illustrated in Figure 8. The executed search path of both agents is shown in Figure 13 for $\tau_{max} = 9$ time steps. Furthermore, the change in total scan certainty ΔS as function of time is shown in Figure 18.

5.3.2 Case 2: Victim-Coverage Conflict

In the second special simulation case, 2 agents are tasked with finding 1 victim in an environment of $(L_x, L_y) = (15, 15)$ which is completely known, save for one square area $E' \subset E$. In this area $c(x, y, \tau_0) = 0$, with $c(x, y, \tau_0) = 1$ elsewhere. This scenario is illustrated in Figure 9. The executed search path of both agents is shown in Figure 14 for $\tau_{max} = 9$ time steps, with the change in total scan certainty ΔS shown in Figure 19.

5.3.3 Case 3: Sensor Radius Exploitation

In this simulation case, 2 agents share the environment of $(L_x, L_y) = (15, 15)$ with 2 victims and a set of obstacles, as is shown in Figure 10. The environment is partially known to the agents, with $c(x, y, \tau_0) = 0.50 \quad \forall x, y \in E$. In this simulation case, a_1 has a sufficiently large perception field such that it can see both victims, whereas a_2 can only see V_1. Furthermore, $h_1(\tau_0) = 20$ and $h_1(\tau_0) = 15$. The executed search paths for both approaches are shown Figure 15 for $\tau_{max} = 10$ time steps. Furthermore, the change in total scan certainty ΔS as function of time is shown in Figure 20.

5.3.4 Case 4: Sensor Accuracy Exploitation

In the fourth case, two agents are tasked with scanning a partially known environment containing two square areas with a lower scan certainty $(E'_1, E'_2) \subset E$, as is shown in Figure 11. The scan certainty values for these areas are as follows:

$$c(x, y, \tau_0) = \begin{cases} 0 & \forall (x, y) \in E'_1 \\ 0.30 & \forall (x, y) \in E'_2 \\ 0.90 & \text{otherwise} \end{cases}$$

Furthermore, Figure 16 shows the paths of both agents for $\tau_{max} = 10$ time steps, and Figure 21 shows the change in scan total certainty as function of time.

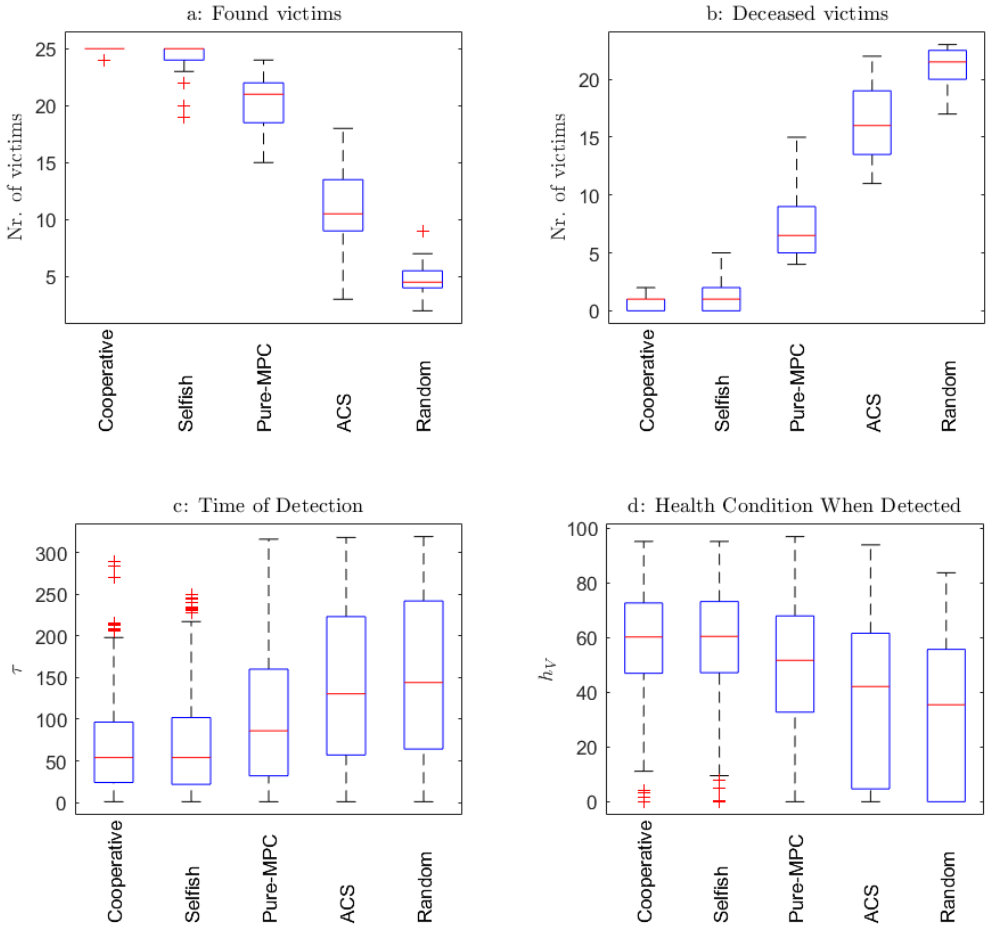

Fig. 7: Victim detection efficiency for each search approach in a random environment, evaluated by (a) the number of found victims, (b) the number of deceased victims, (c) the time of detection, and (d) the health state at the time of detection of a victim.

5.3.5 Combined Simulation Case

Finally, we present a simulation case that combines the various individual scenarios, as described previously. This simulation can be seen in Figure 12 and is bounded by a larger environment of $(L_x, L_y) = (30, 15)$, containing 2 agents and 6 victims. Furthermore, the environment is partially known, with a number of areas $E'_{1-7} \subset E$ of varying initial scan certainty:

$$c(x, y, \tau_0) = \begin{cases} 0.70 & \forall (x, y) \in E'_1 \\ 0.40 & \forall (x, y) \in E'_2 \\ 0.20 & \forall (x, y) \in E'_{3,4,5,6} \\ 0.10 & \forall (x, y) \in E'_7 \\ 0.50 & \text{otherwise} \end{cases}$$

The selfish and cooperative paths of both agents are shown in Figure 17 for $\tau_{max} = 35$. Additionally, Table 6 shows the health state at the end of the simulation, the number of times a victim has been visited (f), and the first time of detection for each victim (τ_f). Similarly to the previous cases, the change in total scan certainty is evaluated and shown in Figure 22.

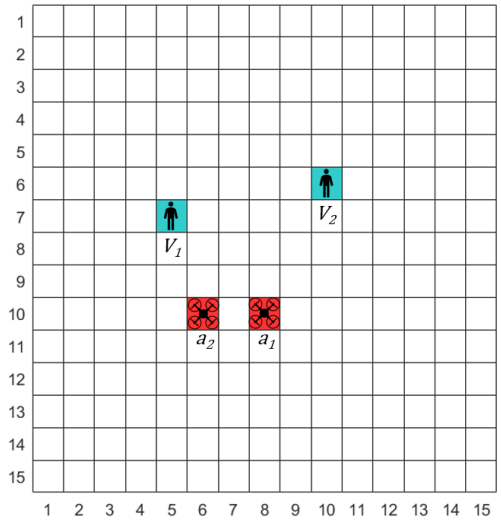

Fig. 8: Special simulation case 1: victim-victim conflict.

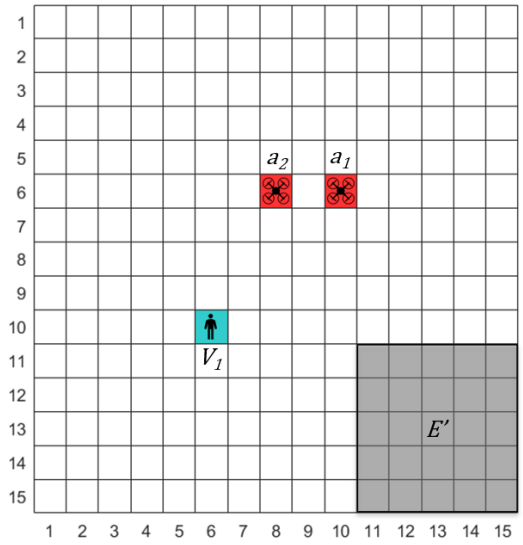

Fig. 9: Special simulation case 2: victim-coverage conflict.

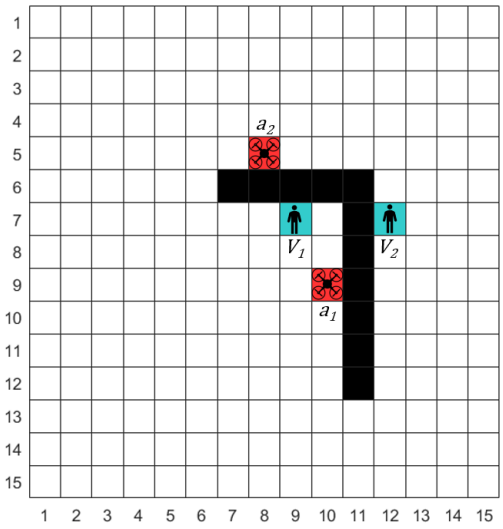

Fig. 10: Special simulation case 3: sensor radius exploitation.

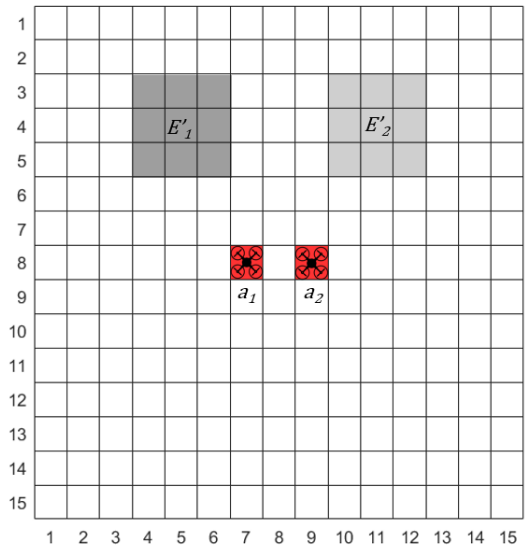

Fig. 11: Special simulation case 4: sensor accuracy exploitation.

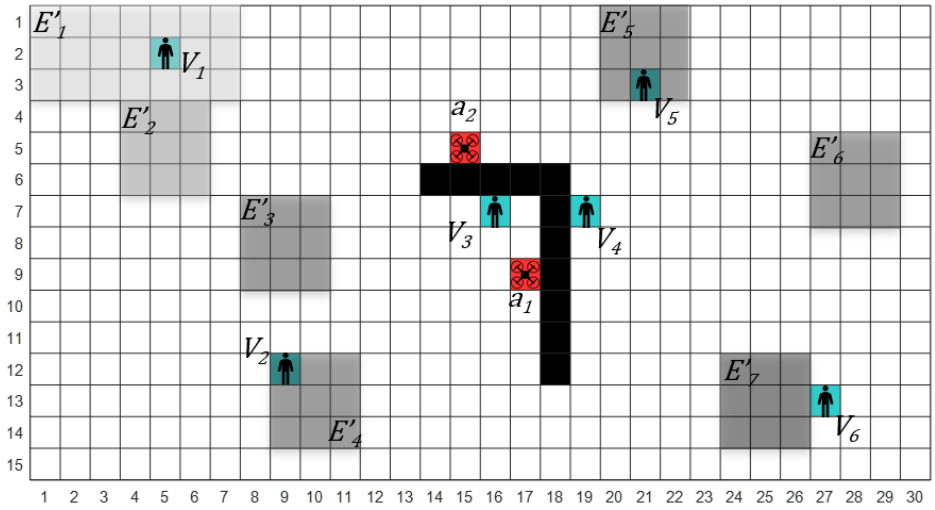

Fig. 12: Combination case simulation.

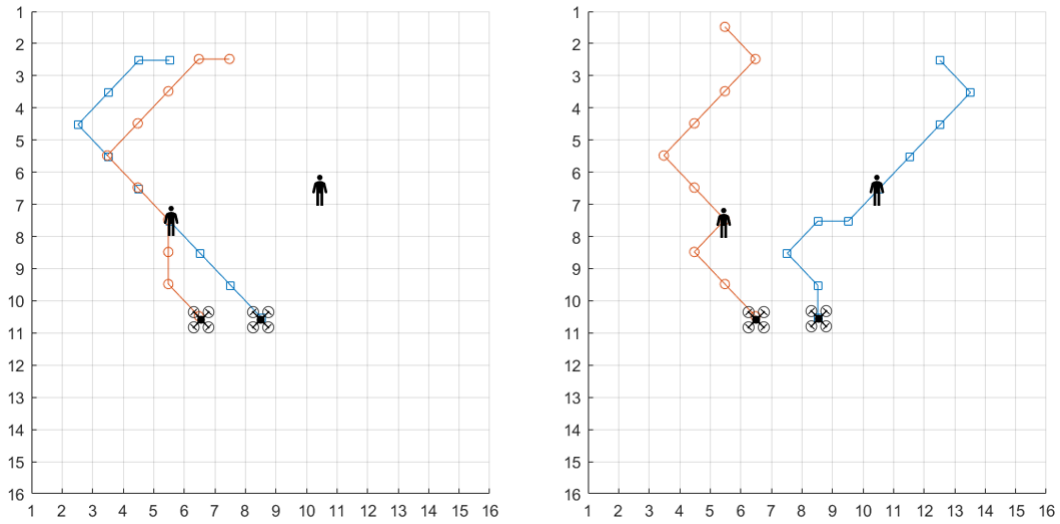

Fig. 13: Case 1 - agent path map using the selfish (left) and the cooperative search approach (right).

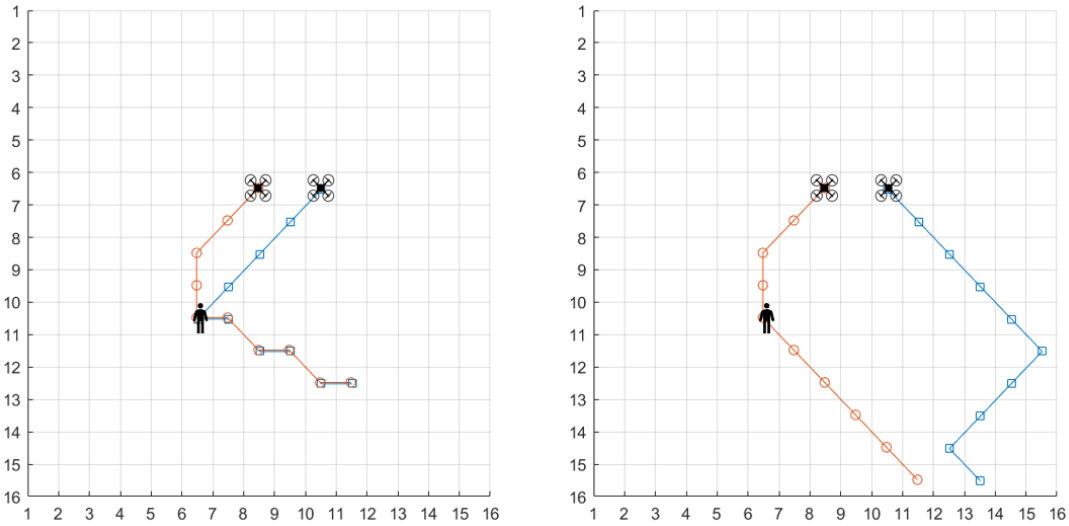

Fig. 14: Case 2 - agent path map using the selfish (left) and the cooperative search approach (right).

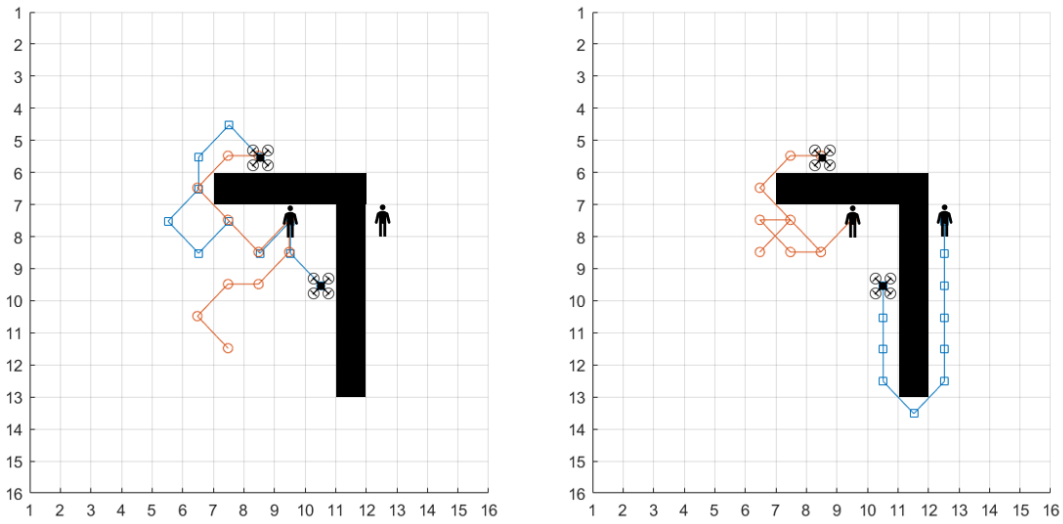

Fig. 15: Case 3 - agent path map using the selfish (left) and the cooperative search approach (right).

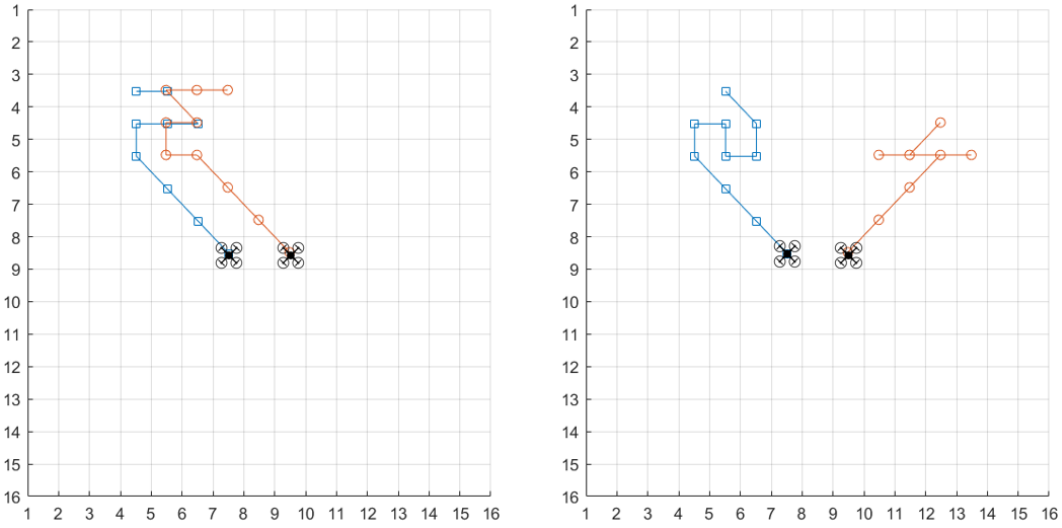

Fig. 16: Case 4 - agent path map using the selfish (left) and the cooperative search approach (right).

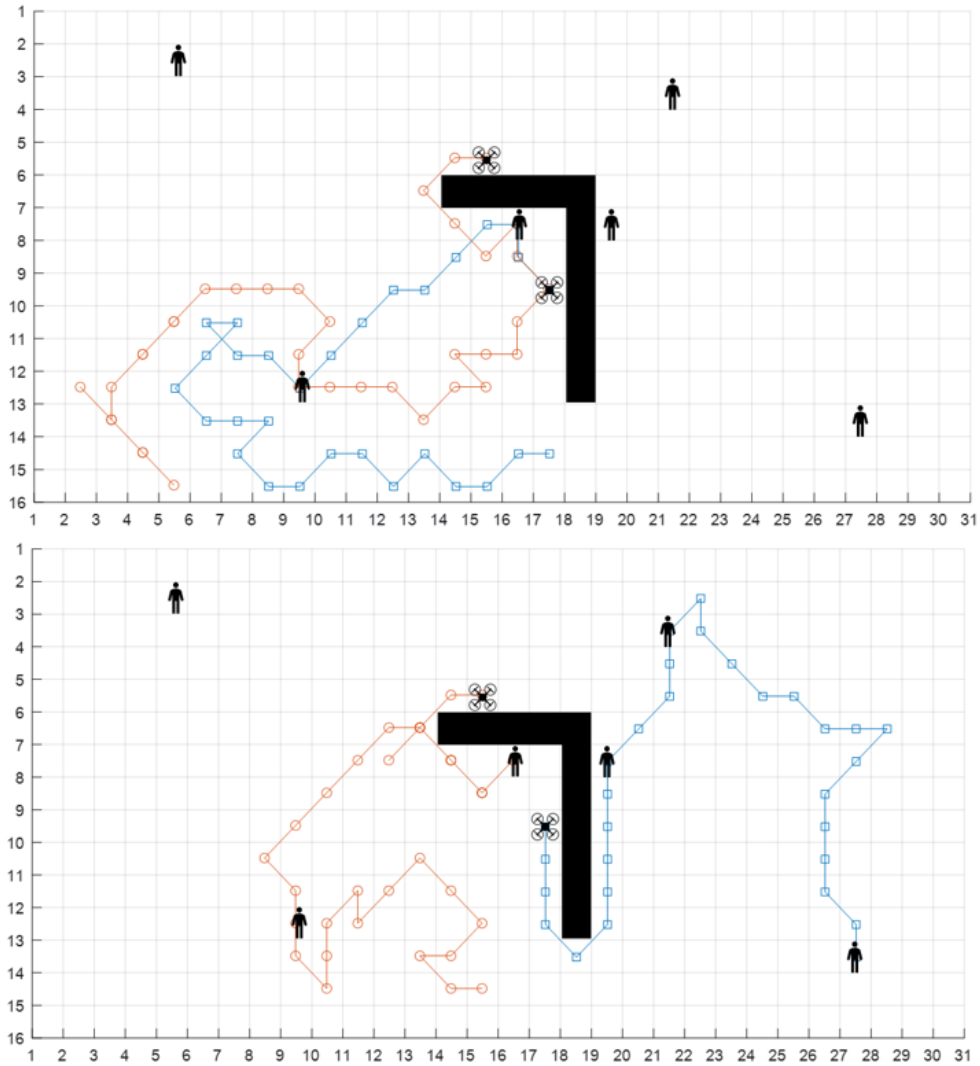

Fig. 17: Combination case - agent path map using the selfish (top) and the cooperative search approach (bottom).

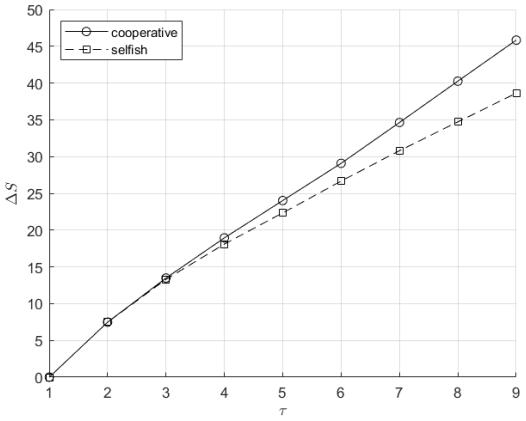

Fig. 18: Case 1 - change in total scan certainty.

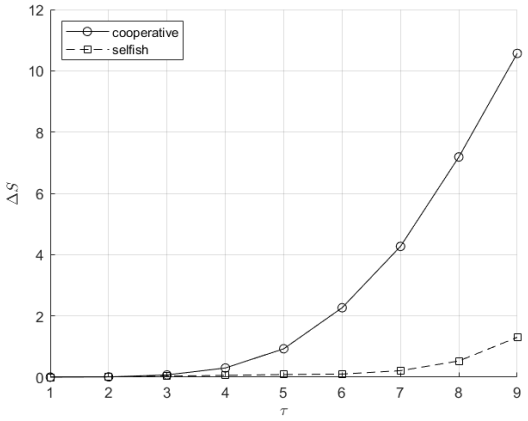

Fig. 19: Case 2 - change in total scan certainty.

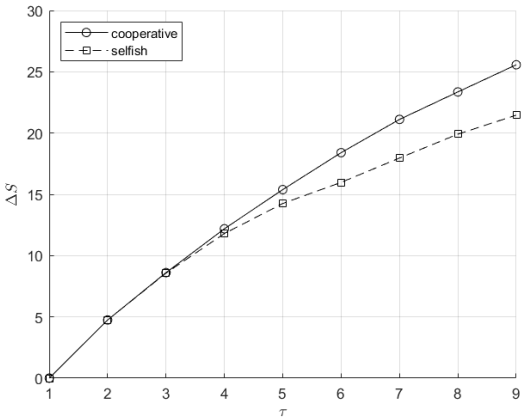

Fig. 20: Case 3 - change in total scan certainty.

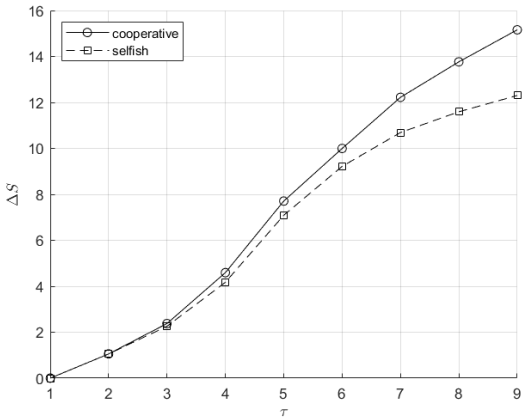

Fig. 21: Case 4 - change in total scan certainty.

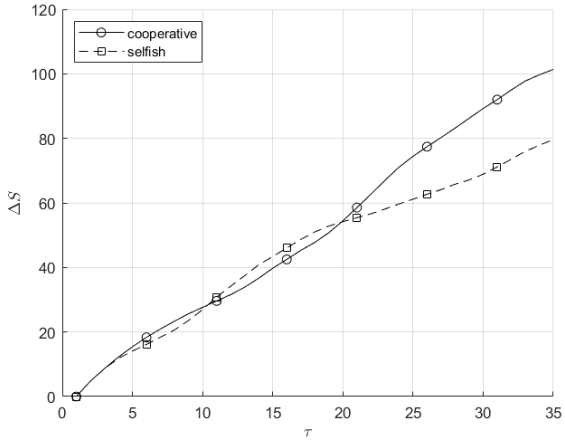

Fig. 22: Combination case - change in total scan certainty.

Table 6: Combination case - victim detection results.

	Selfish			Cooperative			
	$h_v(\tau = 35)$	f	τ_f		$h_v(\tau = 35)$	f	τ_f
V_1	6.94	0	-	V_1	6.94	0	-
V_2	24.9	2	10	V_2	17.2	1	22
V_3	17.97	2	3	V_3	14.34	2	8
V_4	0	0	-	V_4	6.03	1	11
V_5	6.94	0	-	V_5	21.87	1	15
V_6	6.94	0	-	V_6	11.15	1	30

6 Discussion

In this section we interpret and discuss the previously presented results in more depth. From this analysis we draw conclusions about the performance of each search approach.

6.1 Random Environment Simulations

First, we discuss the area coverage performance of all search methods when applied to a random simulation environment. In Figure 5, we show that both the selfish and cooperative search approach achieve a comparable total scan certainty after 300 time steps. However, Table 4 shows that the cooperative approach is significantly faster to reach a particular levels of scan certainty in the later stages of the simulation, for instance achieving 80% total scan certainty 14.4% sooner than the selfish approach. Furthermore, the ACS search approach proves best in terms of final area coverage, with a 6.71% higher $S(300)$ than the cooperative approach, and a comparable rise time in earlier stages. This is likely due to the objective function of the ACS search algorithm, which only uses the scan certainty map to determine the most favourable next step for each agent. Hence, this algorithm has a single, non-conflicting objective, as opposed to the selfish and cooperative approach. The pure-MPC approach, however, performs worse than the cooperative and selfish approach. When investigating the optimisation behaviour of the search approach, it was found that in many cases the solver reaches its fixed iteration limit, and thus does not converge to a globally optimal solution. This shows the added value of warm-starting the supervisory controller with the local controller solution in the cooperative search approach. Lastly, the random search approach achieves the worst degree of area coverage of all approaches, due to its lack of search objective.

Next, Figure 7.a and Figure 7.b show that the cooperative and selfish approach achieve a similar performance in terms of both number of found and deceased victims, with both approaches being able to find all 25 victims in most simulation. However, the cooperative approach has a lower variance for the number of found and deceased victims with $\sigma^2_{\text{f,coop}} = 0.134$ and $\sigma^2_{\text{d,coop}} = 0.345$, respectively. This indicates a more consistent victim search performance compared to the selfish case for which $\sigma^2_{\text{f,self}} = 3.10$ and $\sigma^2_{\text{d,self}} = 2.22$. Yet, both control approaches outperform the pure-MPC, ACS, and random search approach in terms of victim detection. For the ACS search approach, this lower performance can again be explained by the structure of the objective function. Since the ACS algorithm does not consider victim detection in its objective function, the SAR agents will only detect victims they come across by chance, similar to the random search approach. However, the ACS approach still detects more victims than random search, due to the better area coverage performance of the algorithm. Furthermore, the lower area coverage performance of the pure-MPC approach also becomes apparent in its victim detection efficiency, which is again likely due to the convergence issues of the optimisation solver. In Figure 7.c and Figure 7.d it is shown that the cooperative and selfish case perform equally well in terms of victim detection time and the victim health condition at time of detection. This indicates that both search approaches are able to find victims at an equally fast rate, and that neither approach finds these victims in a better or worse health state than the other.

Finally, we analyse the computational effort required for each search approach, as shown in Table 5. Here, both the ACS and random search approach perform the search task in the shortest amount of time, due to their simple and efficient heuristic algorithms. Furthermore, the cooperative approach on average takes 40.5% longer to complete a simulation compared to the selfish search approach. This is due to the additional computations performed by the supervisory controller. This is confirmed by Figure 6, which shows that on average the supervisory controller is activated 41 times when using the cooperative search

approach, with a variance of $\sigma^2_{confl,coop} = 1.56e3$. A similar result can be observed for the pure-MPC approach, which requires the highest computational effort since the supervisory controller is activated at every time step. However, Figure 6 also shows that the SAR agents cause less search conflicts when controlled by the cooperative search approach compared to the selfish approach. From this result we reason that the supervisory controller, when triggered, is able to better prevent future search conflicts between the SAR agents. This strengthens the conclusion that the cooperative search approach is an effective method to spread the SAR agents over the environment.

6.2 Special Simulation Cases

After the random environment simulations, we analyse the results of the special simulation cases starting with Case 1. In the selfish scenario, Figure 13 shows that both agents prioritise V_1 over V_2, since $h_1 < h_2$. Consequently, V_1 is detected by both agents at $\tau = 3$. After this, the agents continue exploring the environment without moving to V_2. With the cooperative approach, both agents are divided over V_1 and V_2, which they find at $\tau = 3$ and $\tau = 5$, respectively. This shows that a SAR system adopting the cooperative search approach is able to successfully coordinate the available agents over multiple victims, where the selfish approach fails. Furthermore, Figure 18 shows that the cooperative approach achieves a ΔS that is 17.2% higher than that of the selfish approach at the end of the simulation. This gain in area coverage is an additional advantage of spreading out the agents over different targets.

Case 2 shows a similar scenario, however now the environment contains two types of potential targets for the agents; one victim and one area of lower scan certainty E'. For the selfish approach, both agents initially head straight to V_1, which they both reach at $\tau = 4$, as shown in Figure 14. After this they both set an identical path for the area with low scan certainty. In the cooperative case, a_1 is tasked with scanning the area with low scan certainty, while a_2 heads to V_1 whom the agent reaches at $\tau = 4$. Both approaches detect V_1 equally fast, however Figure 19 shows that the cooperative approach outperforms the selfish approach with a factor of 4.03 in terms of ΔS. This large difference is a result of the supervisory controller, which successfully identifies both V_1 and E' as targets of interest and divides the available SAR agents between them.

Next, Case 3 shows a situation in which the global mission may strongly conflict with a SAR agent's local objectives. In the selfish case, Figure 15 shows that both a_1 and a_2 locally decide to visit V_1 at $\tau = 2$ and $\tau = 5$, respectively. Although their health state is less critical, V_1 is prioritised over V_2 by a_2 since they can be reached in fewer time steps. Both agents subsequently continue exploring the environment, but are unable to find a feasible path to V_2 due to their limited sensor radius. In the cooperative case, a_1 and a_2 are tasked with finding V_2 and V_1, whom they reach at $\tau = 10$ and $\tau = 5$, respectively. Although a_1 is able to reach V_1 in a shorter time, the supervisory controller in the cooperative control approach decides that it is better to let V_1 be detected at a later time step if that means also detecting V_2. Hence, this simulation case highlights the ability of the cooperative control approach to assign locally sub-optimal tasks to each SAR agent, while still maximising the global mission pay-off. Furthermore, Figure 20 shows the added benefit of quickly spreading out the agents over the environment, where the cooperative search approach achieves a 18.7% higher ΔS compared to the selfish approach.

When analysing Case 4, Figure 16 shows that both agents locally decide to move to E'_1 in the selfish case, since this area is expected to yield the highest gain in scan certainty. With the cooperative search approach, both agents are divided between E'_1 and E'_2. Moreover, a_1 is sent to scan E'_1 for two reasons. First of all, a_1 is closer to this area, which is valued in the objective function of the local controller as explained in Section 4.1.3. Secondly, the supervisory controller exploits the fact that a_1 has a higher sensor accuracy than a_2 and is therefore expected to yield a larger ΔS when sent to this area. In Figure 21 the effect of spreading out the agents in such a manner can be seen in terms of area coverage. By dividing the agents over different search areas, the cooperative search approach achieves a 29.8% higher ΔS compared to the selfish approach.

Finally, the Combination Case integrates elements of the previous special cases in a larger simulation. In the selfish case, both agents locally

decide to visit V_3, after which they move to the southwest quadrant of the environment. Here, they individually visit V_2 and continue to explore the surrounding region. This results in 2 victims detected (of which 2 are visited by both agents), 4 victims undetected, and 1 victim deceased during the simulation. Using the cooperative search approach, a_1 and a_2 are initially tasked with visiting V_4 and V_3, respectively. After they have reached their assigned targets, both agents explore different areas of the environment, which results in V_2, V_5, and V_6 also being found. At the end of the simulation, 1 victim remains undetected, and no victims are deceased. Not only does the SAR system detect more victims when adopting a cooperative search approach, but this approach also ensures that no victim is visited by more than one different agent. This shows that the cooperative search approach performs better in terms of victim search efficiency. Moreover, by dividing the agents over different victims, the SAR system is able to simultaneously visit more areas with a reduced scan certainty using the cooperative control approach. This becomes apparent when evaluating the change in total scan certainty as function of time, as shown in Figure 22. At the end of the simulation, the SAR system achieves a 27.6% higher ΔS when adopting a cooperative search approach, as opposed to the selfish control approach.

7 Conclusions & Topics for Future Research

In search-and-rescue missions, it is imperative to create situational awareness of an unknown environment. To achieve this, multi-agent systems of cooperative robots can be deployed for fast and efficient area mapping. This paper has presented the framework for a new type of hierarchical, cooperative mission planning control approach for the application of multi-agent search-and-rescue systems. Moreover, this control approach incorporates non-homogeneous sensory capabilities of the SAR agents in its search strategy, in order to increase its victim detection efficiency and area coverage performance. On a lower level, each SAR agent has its own heuristic local controller, which uses FLC and a k-shortest path algorithm to determine its own target and path. At a higher level, a centralised MPC-based supervisory controller coordinates the global mission of all SAR agents if two or more agents have conflicting search objectives. The rationale is that the distributed local controllers can, in most cases, efficiently find a satisfactory, sub-optimal solution to guide each SAR agent individually at a given time step. In the less frequent case of a search conflict, the centralised supervisor can optimise the global pay-off for all agents, at the cost of being more computationally demanding. This control architecture is termed the *cooperative* search approach, as opposed to the *selfish* approach, in which the SAR agents are only guided by their own local controller.

In simulations with randomly positioned obstacles and victims, the cooperative and selfish search approach show similar performance in terms of victim detection efficiency and area coverage, although the cooperative approach is faster at reaching higher levels of area scan certainty than in the selfish case. However, it is shown that in specific SAR situations the cooperative approach greatly outperforms the selfish approach by efficiently resolving various types of search conflicts. Furthermore, the area coverage performance of the cooperative approach is comparable to a purely heuristic approach designed for area coverage, while simultaneously outperforming this approach in victim detection efficiency. Similarly, the cooperative search approach outperforms a purely optimisation-based approach in both area coverage and victim detection efficiency, showing that the hierarchical combination of the local and supervisory controller performs better than either one control type separately. This comes at the cost of the cooperative approach being more computationally expensive than the selfish, ACS, and random search approach. However, we deem this acceptable and necessary when considering the higher-ranking performance of this search approach in both victim detection efficiency and area coverage.

We identify a number of topics for future research. First, the fidelity of victim and agent simulation can be improved by using more extensive models for their behaviour, movement, and physical capabilities/limitations. For example, this research only considers non-homogeneous sensory capabilities between the SAR agents. However, this can be extended to differences in moving

speed, computational capacity, or task specialisation (such as area mapping, victim aid, explosive removal, etc.). Moreover, this recommendation can be extended to the simulation environment itself. True SAR scenarios include a wide variety of (dynamic) obstacles and hazards that need to be considered by the mission planning controller. Future research may include the modelling of fire spread, smoke, explosive materials, and falling debris, which require increasingly robust controllers. Furthermore, we recommend further research into the effects of increasing the SAR fleet size. It is hypothesised that the performance differences between search approaches will become more pronounced for an increased SAR system fleet size, albeit at the cost of more computational effort for the supervisory controller. Investigating the extent of this effect can contribute to quantifying the feasibility and real-life applicability of this research.

References

[1] Casper, J., Murphy, R.R.: 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) **33**(3), 367–385 (2003)

[2] Coburn, A.W., Spence, R.J.S., Pomonis, A.: Factors determining human casualty levels in earthquakes: mortality prediction in building collapse. In: Proceedings of the Tenth World Conference on Earthquake Engineering, vol. 10, pp. 5989–5994 (1992). Rotterdam, Netherlands

[3] Riley, J.M., Endsley, M.R.: The hunt for situation awareness: Human-robot interaction in search and rescue. In: Proceedings of the Human Factors and Ergonomics Society Annual Meeting, vol. 48, pp. 693–697 (2004). Los Angeles, CA

[4] Shimanski, C.: Situational awareness in search and rescue operations. In: International Technical Rescue Symposium (2005)

[5] Liu, Y., Nejat, G.: Robotic urban search and rescue: A survey from the control perspective. Journal of Intelligent & Robotic Systems **72**(2), 147–165 (2013)

[6] Grogan, S., Pellerin, R., Gamache, M.: The use of unmanned aerial vehicles and drones in search and rescue operations - a survey. In: Proceedings of the PROLOG (2018). Hull, UK

[7] Beck, Z., Teacy, W.L.T., Jennings, N.R., Rogers, A.C.: Online planning for collaborative search and rescue by heterogeneous robot teams. In: Proceedings of the International Conference on Autonomous Agents & Multiagent Systems, pp. 1024–1033 (2016). Singapore

[8] de Alcantara Andrade, F.A., Reinier Hovenburg, A., Netto de Lima, L., Dahlin Rodin, C., Johansen, T.A., Storvold, R., Moraes Correia, C.A., Barreto Haddad, D.: Autonomous unmanned aerial vehicles in search and rescue missions using real-time cooperative model predictive control. Sensors **19**(19), 4067 (2019)

[9] San Juan, V., Santos, M., Andújar, J.M.: Intelligent uav map generation and discrete path planning for search and rescue operations. Complexity **2018** (2018)

[10] Yao, P., Zhao, Z.: Improved glasius bio-inspired neural network for target search by multi-agents. Information Sciences **568**, 40–53 (2021)

[11] Galceran, E., Carreras, M.: A survey on coverage path planning for robotics. Robotics and Autonomous systems **61**(12), 1258–1276 (2013)

[12] Koenig, S., Liu, Y.: Terrain coverage with ant robots: a simulation study. In: Proceedings of the Fifth International Conference on Autonomous Agents, pp. 600–607 (2001). Montreal, Canada

[13] Wagner, I.A., Altshuler, Y., Yanovski, V., Bruckstein, A.M.: Cooperative cleaners: A study in ant robotics. The International Journal of Robotics Research **27**(1), 127–151

(2008)

[14] Yang, S.X., Luo, C.: A neural network approach to complete coverage path planning. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) **34**(1), 718–724 (2004)

[15] Yang, Y., Polycarpou, M.M., Minai, A.A.: Multi-uav cooperative search using an opportunistic learning method. Journal of Dynamic Systems, Measurement, and Control **129** (2007)

[16] Arnold, R., Jablonski, J., Abruzzo, B., Mezzacappa, E.: Heterogeneous uav multi-role swarming behaviors for search and rescue. In: IEEE Conference on Cognitive and Computational Aspects of Situation Management, pp. 122–128 (2020). Victoria, BC, Canada

[17] Choi, H., Brunet, L., How, J.P.: Consensus-based decentralized auctions for robust task allocation. IEEE Transactions on Robotics **25**(4), 912–926 (2009)

[18] Khamis, A.M., Elmogy, A.M., Karray, F.O.: Complex task allocation in mobile surveillance systems. Journal of Intelligent & Robotic Systems **64**(1), 33–55 (2011)

[19] Elston, J., Frew, E.W.: Hierarchical distributed control for search and tracking by heterogeneous aerial robot networks. In: IEEE International Conference on Robotics and Automation, pp. 170–175 (2008). Pasadena, CA, USA

[20] Chandler, P.R., Pachter, M., Rasmussen, S.: Uav cooperative control. In: Proceedings of the American Control Conference, vol. 1, pp. 50–55 (2001). Arlington, VA, USA

[21] Wang, W., Joshi, R., Kulkarni, A., Leong, W.K., Leong, B.: Feasibility study of mobile phone wifi detection in aerial search and rescue operations. In: Proceedings of the 4th Asia-Pacific Workshop on Systems, pp. 1–6 (2013). Singapore

[22] Ganesan, S., Shakya, M., Aqueel, A.F., Nambiar, L.M.: Small disaster relief robots with swarm intelligence routing. In: Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief, pp. 123–127 (2011). Kollam, India

[23] Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE transactions on Systems Science and Cybernetics **4**(2), 100–107 (1968)

[24] Dijkstra, E.W., *et al.*: A note on two problems in connexion with graphs. Numerische mathematik **1**(1), 269–271 (1959)

[25] Yen, J.Y.: Finding the k shortest loopless paths in a network. Management Science **17**(11), 712–716 (1971)

II

Literature Study

Executive Summary

Both natural and man-made disasters claim many lives and cause immeasurable societal damage around the world. Due to the unpredictable nature of all disasters, there will always be a need for more efficient search-and-rescue (SAR) protocols in response to a disaster. This literature study aims to support a thesis research into reducing victim search time of a multi-agent search-and-rescue system (SARS) of non-homogeneous unmanned aerial vehicles (UAVs) in the response and relief phase of a disaster. Specifically, the thesis proposal aims to contribute to the research field of SARS, by developing a mission planning approach which will exploit the non-homogeneous nature of each UAV in the system, thus increasing the victim search efficiency of the system. This literature review first covers various types of control approaches relevant to SAR mission planning, including fuzzy logic control (FLC), prioritised planning, and model-predictive control (MPC). For each control approach, the fundamental principles, characteristics, and common fields of application are explained, along with relevant extensions or adapted variations of the control methods. This is followed by a critical review of the state-of-the-art of current SARS research. The literature is broken down and discussed based on disaster environment, fleet composition, sensory capabilities of the agents, task assignment, path planning, and the uncertainties and disturbances that are incorporated in the SARS research. From this review, a number of trends can be identified in the literature. The first of these trends, is that airborne robots have taken over land-based robots as the dominant type of robotic SARS agent in the past decade. Furthermore, mathematical control approaches for mission planning are becoming popular in SARS research, although heuristic approaches are still considered viable for current applications. Moreover, sensors that can detect wireless devices have been found viable for victim detection in a SAR setting, in combination with (more conventional) sensors such as cameras. The final trend, is that uncertainty and disturbances in the SAR environment are rarely explicitly addressed in (theoretical) SARS research, leaving room for extensions and improvements on such investigations. From the reviewed literature on control approaches, SARS research, and SAR simulator design, it is possible to formulate this thesis research proposal. First, a novel control approach termed *Fuzzy-Logic Assisted Prioritised Task Assignment and Path Planning* (FLAPTAPP) is proposed for the mission planning aspect of the SARS. The approach combines both task assignment and path planning in one controller, utilising fuzzy logic control (FLC) and prioritised planning, respectively. This mission planning approach is designed to exploit the individual capabilities of the agents in the SARS, thus contributing to non-homogeneous SARS mission planning. Additionally, the world model that the mission planning controller uses, is updated in real-time to improve the search strategy adopted by the mission planning module. Finally, to validate the proposed SARS of this thesis work, a custom simulator is built based on both previous work by the research group of this thesis work, and the internationally recognised SAR environment simulator by Robocup-Rescue. The contributions of this research to the field of SARS, are both a more extensive mission planning approach which exploits the individual capabilities of the agents in the SARS, and a novel controller type, that combines fuzzy task assignment and prioritised path planning.

1

Introduction

Disasters frequently grip the headlines of news media. Such articles may be focused on a single cataclysmic event, such as the 2020 Beirut explosion, which resulted in at least 204 lives lost, 6500 injured, an estimated 300,000 citizens left homeless, and US$15 billion in damage [1][2]. However, certain types of disasters can be a more regular (even seasonal) occurrence, such as climate-related and geophysical disasters. Based on an investigation in [3], publicised by the United Nations Office for Disaster Risk Reduction (UNDRR), these types of disasters accounted for 1.3 million casualties in the period between 1998 and 2017, with a further 4.4 billion directly affected[1] by the event.

In both the single and reoccurring case, these statistics show that disasters can have an enormous impact on society and the environment. Disaster management (also referred to as emergency management) involves the preparation for, response to, and recovery after a disaster. In the context of disaster management, the UNDRR defines a disaster as follows[2]:

> *"[A disaster is] 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."*
>
> (UNDRR, 2020).

In [4], Khorram et al. provide a detailed review of many different aspects of disaster management, which corresponds with the vision of the UNDRR. Most notably, they state that an important characteristic of disasters is that they cannot be prevented. Floods, fires, or pandemics are almost always unavoidable events, of which only the consequences can be mitigated. This stresses that, no matter how rigorous the preparation, there will always be a need for efficient response and recovery protocols to soften the aftermath of a disaster. One task within the response and recovery phase of any disaster, is victim aid and relief. For this objective, rescue teams are deployed in a disaster scenario.

1.1. Search-and-Rescue Robotics

In the past, a search-and-rescue system (SARS) consisted of almost exclusively human rescue workers (with the exception of trained rescue animals). The past couple of decades have seen a rapid increase in the applications of robots in the field of urban search-and-rescue (USAR) disaster settings. Recent advances in search-and-rescue (SAR) robots are motivated by a number of advantages of these systems over teams of only human workers. From a safety perspective, the deployment of (autonomous) robots reduces the risks that human rescue workers would be subject to. For example, a system of crawlers was deployed in the direct aftermath of the World Trade Centre attack on 11 September 2001, operating in environments with risks such as compromised structures, fire, and (toxic) dust and gasses [5]. The application of robots stems from the philosophy that machines are expendable and human lives are not. Therefore, human rescue workers no longer need to risk their own lives to save that of others.

[1] Injured, left homeless or displaced, or in need of emergency assistance.
[2] https://www.undrr.org/terminology/, (Retreived November 2020)

Furthermore, human resources can be scarce in a USAR scenario, and the automation of search and relief tasks by means of autonomous robots reduces the workload on human rescue personnel. This allows that vital human resources can be made available for other tasks such as logistics and victim aid. Apart from safety and resource availability, autonomous robots can offer another advantage over human workers in the form of victim detection efficiency. In the initial period directly after any disaster, search time is a critical factor when it comes to recovering survivors. Coburn et al. in [6] propose a model for mortality predictions after an earthquake. Here, they determine that survival rates of victims greatly increase with increased rescue efficiency in the first 24-36 hours after the disaster. In an effort to increase this victim detection efficiency (and thus reduce search time), fast, agile autonomous drones such as unmanned aerial vehicles (UAVs) are deployed to map out disaster areas faster which might be entirely inaccessible to humans and ground robots.

For faster response and relief, a SARS can be combined with more efficient autonomous mission planning approaches. For this thesis research, mission planning is defined as the combination of two problems. First, the problem of assigning a task or destination to an agent, and secondly, to determine a corresponding path to reach it. Task assignment becomes a factor in mission planning when there are more tasks than agents in a multi-agent system. In other words, the mission planner must decide which tasks get prioritised over others. Furthermore, the task assignment problem acquires an extra dimension when the SARS has non-homogeneous agents. A homogeneous system is characterised by all agents being roughly identical; they have the same performance capabilities (range, speed, computational power), the same payload/sensors, the same functionality, and thus essentially the same relative value. However, in a non-homogeneous multi-agent system, different agents may have different capabilities, and as a result may be more specialised or efficient for different tasks within the SAR mission. Not only does the mission planning module decide on the priority of each area in the search grid, but it must also determine which agent's capabilities are best suited for each task. The second aspect of mission planning is determining the path an agent should take to reach its destination, also referred to as trajectory planning. In a disaster environment with varying dynamics and uncertain elements (moving victims, spreading fire, compromised structures), it can be challenging to find the fastest feasible path to a particular destination. What adds to the challenge is the fact that a mission planning module must run constantly in real-time, in order to keep updating the commands that the SARS agents receive. These challenges of both task assignment and trajectory planning, combined with the requirements imposed on the SARS, raise the need for more advanced mission planning approaches.

1.2. Project Scope

This literature study is meant to support a thesis research into reducing victim search time of a multi-agent SARS of non-homogeneous UAVs in the response and relief phase of a disaster. The thesis centres around designing a mission planning controller, with the goal to minimise victim search time and maximise the area coverage of an urban disaster setting. More specifically, the mission planner is designed for a non-homogeneous SARS, and strongly focuses on exploiting the different capabilities of the system's agents. Since the mission planning module will incorporate the specific capabilities of each agent into its strategy, it is hypothesised that such an approach will increase the victim search efficiency of the SARS with respect to both the search time and area coverage. The mission planning controller will be adaptive, since its strategy is dependent on input data that the agents gather during their deployment, such as unexpected fire hazards or damaged structures. Essentially, the controller will progressively 'learn' more about the environment, which it will use to further optimise its search strategy.

The thesis research will consist of two main parts. First, the control approach for mission planning is developed. This is followed by implementing the mission planning approach in a simulation environment. In this simulation, the mission planning module will be applied to a SAR environment and evaluated based on its performance.

1.3. Project Objectives

The ultimate goal of any SAR mission is to save as many lives as possible, and to do this in the least possible time. From this, it follows that the main research objective of this thesis is:

"To reduce the total number of casualties caused by a disaster, by increasing the victim

search efficiency with respect to mission time and area coverage of a multi-agent, non-homogeneous search-and-rescue system of UAVs."

This main objective of the research can be divided into a number of sub-goals. The main contribution of this research will focus on optimising the victim search behaviour of a SARS with non-homogeneous agents, with a mission planning controller that uses dynamic models of the environment for its decision making. Since there are many possible control methods for both trajectory planning and task assignment, an appropriate control method must first be chosen before it can be designed. This will form the first sub-goal of the research. The environment model that the resulting mission planning controller uses, is constantly updated based on the information gathered by the agents in the disaster scene. Hence, the mission planning approach will not only exploit the individual capabilities of non-homogeneous agents, it will also have an adaptive structure to account for changes in the disaster environment. It follows that one sub-objective of this research will be to keep updating the environment model for the mission planner, by combining it with real-time input data provided by the agents. The mission planning controller will provide information/input for the agents to steer their search behaviour. Hence, another sub-objective will be to set up appropriate information exchange between the mission planning controller and agents, by investigating what information they require for optimal mission planning. Furthermore, it is a requirement to run the mission planning module in real-time. If the computational load is found to be too high for this application, it must be mitigated by reducing and/or distributing the load. Finally, it is necessary to test and evaluate the resulting system. For this, the victim search efficiency of the SARS will be tested, by designing a simulation environment with all necessary elements, including fire, structural damage, victim condition and behaviour, and wind.

1.4. Research Questions

From the objectives of this thesis, it is possible to formulate the main research question and its sub-questions. The main research question of this thesis is:

"Can an adaptive, autonomous mission planning approach for non-homogeneous agents increase the victim search efficiency of a robotic search-and-rescue system?"

To support this research question, a number of sub-questions have been formulated. Answering these questions will form the basis for achieving the sub-goals of this research, which have been discussed in Section 1.3.

- What control approach is best suited for non-homogeneous, multi-agent mission planning in terms of victim search time and area coverage?
- How can an adaptive mission planning approach be used to incorporate real-time data from the agents in the system?
- What information exchange is required between agents in a multi-agent SARS?
- What information exchange is required between the agents and the mission planning module in a multi-agent SARS?
- How can the computational load of a mission planning controller be managed, such that it can be solved at each time step in the system in real-time?
- How should a simulation environment be designed in order to assess the victim search efficiency of a SARS?

1.5. Literature Study Structure

This literature review is structured as follows. First, Chapter 2 discusses the theory, characteristics, and various applications of a number of control methods. This provided the basis for determining which control approach may be best suited for the mission planner. This is followed by Chapter 3, in which the specifics of mission planning and agent sensing are addressed. Additionally, this chapter includes a brief overview of recent SARS researches and evaluate them based primarily on mission planning,

fleet composition, and the communication and sensing infrastructure of the agents. Next, Chapter 4 reviews and compares different modelling sources en methods for the simulation environment, in which the developed mission planning approach are tested. Once the mission planner and simulation environment are reviewed, the final proposed system for the thesis research is summarised in Chapter 5. Finally, Chapter 6 concludes this literature study and recapitulates the key points of all chapters.

2

Control Approaches

Before research on different SARS applications can be reviewed and compared, it is necessary to provide a theoretical basis for the commonly used control approaches. This chapter reviews a number of heuristic and mathematical control methods by discussing the main concepts, controller characteristics, and relevant theoretical and practical extensions of the main theory.

The first control approach is fuzzy logic control, discussed in Section 2.1. This is followed by a review of model-predictive control (MPC) in Section 2.3, and prioritised planning in Section 2.2. Next, a number of hybrid control approaches are reviewed in Section 2.4, which are combinations or extensions of the previously discussed control methods. Finally, Section 2.5 presents a concise summary of the main characteristics of each controller reviewed in this chapter.

2.1. Fuzzy Logic Control

This section provides an overview of fuzzy logic and its application in fuzzy logic control (FLC). Although the philosophical idea of fuzzy theory has existed for far longer, it was Zadeh who first introduced it as a mathematical discipline in 1965 in [7]. Additionally, Dubois presents a broad, complete overview of fuzzy logic and fuzzy systems in [8]. These sources will form the main basis for the following review of FLC.

In Section 2.1.1, a brief introduction is given to fuzzy logic as a mathematical concept. This is followed by an overview of different types of rule-based fuzzy systems and their corresponding inference mechanisms in Section 2.1.2. Finally, Section 2.1.3 discusses different forms of fuzzy systems in control applications, along with a number of research examples.

2.1.1. Fuzzy Logic

In general, fuzzy logic is a method for representing (non-stochastic) uncertainties in a mathematical framework and for mathematical reasoning in presence of uncertainty. To better understand this, first the basic elements of fuzzy logic are explored, starting with *fuzzy sets*. In classical set theory, sets are crisp or well-defined, i.e. an object x is either a member of a set A, or not a member of the set. In mathematical terms, its so-called membership (or characteristic) function is equal to 1 or 0, as is shown in Equation (2.1):

$$\mu_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \tag{2.1}$$

However, there exist more vague, ambiguous, or common-sense concepts for which sets are not well-defined. For instance, when defining *"the set of very tall people"* or *"the set of expensive cars"* it might not always be clear whether an element belongs to it. This is due to the fact that linguistic, 'human' terms are used to define the sets, instead of exact, quantified terms. However, these types of ambiguous concepts are still an essential part of human reasoning, decision making, and (manual) control. To define such uncertainties mathematically, fuzzy sets have been developed. Contrary to classical sets, an object can be a partial member of a fuzzy set, as shown in Equation (2.2). Instead

of being a binary relation, the membership function $\mu_A(x)$ is now a mapping of X on the unit interval $\mu_A(x) : X \rightarrow [0, 1]$.

$$\mu_A(x) = \begin{cases} 1 & \textit{x is with full certainty member of A} \\ \in (0,1) & \textit{x is partially a member of A} \\ 0 & \textit{x is with full certainty not member of A} \end{cases} \tag{2.2}$$

As an example, the *"set of very tall people"* can be considered. The linguistic term *'very tall'* is a vague and subjective measure which humans use to describe their height, since it is inexact and dependent on one's personal interpretation of the terms. For instance, one's own height or sex might strongly influence their definition of a very tall person. In this case, it is convenient to first define a function that corresponds a membership degree to a certain realisation of the linguistic terms. For example, one can argue that anyone shorter than 1.85m is definitely not very tall, whereas anyone taller than 2.10m definitely is. The area in between the two limits is where people have a partial (fuzzy) membership to the *"very-tall-people-set"*. When graphically displaying the resulting membership function, it could look like Figure 2.1.

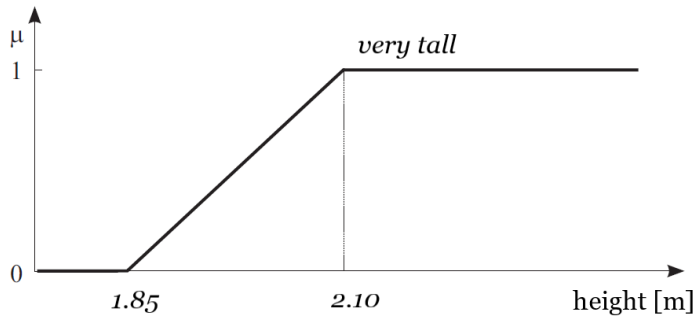

Figure 2.1: Membership function corresponding to the set of very tall people

2.1.2. Rule-Based Fuzzy Systems

Fuzzy systems are built on information originating from a-priori knowledge and/or data measurements. A-priori knowledge is usually gathered from system experts, such as process engineers or designers. Broadly speaking, a fuzzy system is any system that uses fuzzy sets and their mathematical principles. Fuzzy sets can be present in (control) systems in a number of different ways, however this research primarily focuses on systems with fuzzy sets in the description of the system. This can, for instance, be in the form of a set of if-then rules with fuzzy propositions. Such systems are called *rule-based fuzzy systems*, and will form the main interest of this literature review due to their wide range of applications in controllers. The rules of a rule-based fuzzy system have the following structure:

If *antecedent proposition* **then** *consequent proposition*.

In such systems the antecedent is always fuzzy, whereas the consequent can be a number of different types, such as a crisp value, a function, or another fuzzy set. The knowledge of the process experts is captured in these if-then rules instead of mathematical formulas. The rationale is that such a rule base can mimic the behaviour of an experienced system operator. Two main types of rule-based fuzzy models are discussed in this literature review, which are the *linguistic fuzzy model* (and *singleton fuzzy model*) and the *Takagi-Sugeno model*. Furthermore, in control theory, fuzzy systems are used to map input-output relations. In rule-based fuzzy systems, this is done by means of *inference*, which is the process of deriving an output fuzzy set given the rule basis and the input. For the previously mentioned fuzzy systems, the most common inference methods are discussed as well.

Linguistic Fuzzy Model

The if-then rules of a linguistic fuzzy model are formulated in the following structure:

R_i: **If x** is A_i **then y**$_i$ is B_i i = 1,2,... K

In these fuzzy models, both the antecedent and consequent are fuzzy. There are many different inference methods to obtain fuzzy outputs from inputs and a given rule base. One of the most widely used inference methods is Mamdani Inference, named after the mathematician who first applied it to the control of a steam engine in [9]. The general steps of the Mamdani inference algorithm are shown in Algorithm 1. First, the input x is converted to a fuzzy set A'. This step is also referred to as *fuzzification*. Next, the degree of fulfilment β of the input is calculated for each rule. Essentially, β resembles how well the input matches with the antecedent membership function of each rule. This is done by taking the maximum of the intersection of the input membership function $\mu_{A'}(x)$ and the membership function of each rule in the rule base $\mu_{A_i}(x)$. In the second step, the individual fuzzy consequent sets are derived, by taking the intersection of each degree of fulfilment with the consequent membership function of each rule. Finally, all consequent sets are aggregated into one fuzzy set B', by finding the maximum of all output membership functions at each output value y.

In most control applications, a fuzzy output set must be converted into a crisp, numerical value which can be processed by a computer. This extra conversion is called *defuzzification*. A common defuzzification method is the centre-of-gravity (COG) method, in which the weighted average of the membership function corresponding to the consequent fuzzy set is taken as the crisp output. The equation for the COG method is shown in Equation (2.3). Another defuzzification approach is the mean-of-maxima (MOM) method, in which a crisp output is obtained by taking the average value of the arguments for which the fuzzy set is at its maximum. The MOM relation is shown in Equation (2.4), where Y is the domain of output y. If there is only a single maximum in the consequent fuzzy set, the MOM method can greatly save computational effort compared to the COG method. However, choosing the appropriate defuzzification method strongly depends on the nature and application field of the fuzzy system itself.

Algorithm 1: Mamdani Inference

1. Fuzzify the input ;
   $A' = \{\mu_{A'}(x)/x \quad | x \in X\}$;
2. Compute the degree of fulfilment of the input for each rule in the rule base ;
   $\beta_i = max_x[\mu_{A'}(x) \wedge \mu_{A_i}(x)] \qquad 1 \leq i \leq K$;
3. Derive the fuzzy sets of the output for each rule ;
   $B_i' : \mu_{B_i'}(y) = \beta_i \wedge \mu_{B_i}(y) \qquad 1 \leq i \leq K$;
4. Aggregate the output fuzzy sets into one ;
   $B' : \mu_{B'}(y) = \max\limits_{1 \leq i \leq K} \mu_{B_i'}(y)$

$$y'_{COG} = COG(B') = \frac{\sum_{i=1}^{N} \mu_{B'}(y_i) y_i}{\sum_{i=1}^{N} \mu_{B'}(y_i)} \tag{2.3}$$

$$y'_{MOM} = mean\{\hat{y} | \quad (\mu_{B'}(\hat{y}) = \max_{\hat{y} \in Y} \mu_{B'}(\hat{y}))\} \tag{2.4}$$

A special case within linguistic fuzzy models, is the *singleton fuzzy model*. In a singleton fuzzy model, the consequence propositions of the rule base are real constants (also called singleton sets). In such a model, the rule base has the following structure:

R_i: **If x** is A_i **then** y_i is b_i i = 1,2,... K

For singleton fuzzy models, the Mamdani inference algorithm described in Algorithm 1 can still be used, although it can be simplified considerably after step 2. Now, $\mu_B(y) = 1$ for $y = y_0$ and $\mu_B(y) = 0$ for all other values of y. This means that the the COG-method for defuzzification can directly be combined with the computed degrees of fulfilment to calculate a crisp output, as shown in Equation (2.5).

$$y' = \frac{\sum_{i=1}^{N} \beta_i b_i}{\sum_{i=1}^{N} \beta_i} \tag{2.5}$$

Takagi-Sugeno Fuzzy Model
The second type of rule-based fuzzy model is the Takagi-Sugeno (TS) fuzzy system. This fuzzy system type is named after the mathematicians who first developed it and applied it to a water cleaning plant and a converter in a steel-making plant [10]. In TS rule bases, the consequent is a function of the antecedent variables, instead of being a fuzzy proposition. This does require the antecedent variables to be crisp values. The rule structure of a TS fuzzy system is as follows:

R_i: **If x** is A_i **then** $\mathbf{y}_i$ is $\mathbf{f}_i$**(x)** i = 1,2,... K

The inference approach for a TS model is a simple extension of the inference method used for singleton fuzzy models, discussed in Equation (2.5). Similarly, an average of the consequent function values is taken, weighted with how well the membership function of the input fits each of the antecedent functions (i.e. the degrees of fulfilment). This yields a crisp output value y', as is shown in Equation (2.6):

$$y' = \frac{\sum_{i=1}^{N} \beta_i \mathbf{f}(\mathbf{x}_i)}{\sum_{i=1}^{N} \beta_i} \tag{2.6}$$

2.1.3. Fuzzy Logic Controllers

As mentioned previously, rule-based FLC aims to mimic the adaptive, non-linear decision making behaviour of human operators by integrating process knowledge in its rule base. Since this knowledge is generally very qualitative, FLC-based approaches are heuristic by nature. Hence, FLC can be a very efficient method for control input estimation, at the cost of being non-optimal approaches. Moreover, apart from being able to perform non-linear control tasks, FLC has a very transparent logic structure. This is an advantage when designing or adjustments in FLC, since a designer can readily find the different sources of knowledge and reasoning integrated in the rule base. In more black-box approaches, such as in artificial neural networks (ANNs), a lack of such transparency may lead to a poor understanding of the underlying process. There are different ways to implement FLC in processes. Three categories are distinguished for the purpose this research:

- **Direct fuzzy logic control**; the reference and feedback variables form the input to a fuzzy logic controller, which directly commands a control output.
- **Supervisory fuzzy logic control**; a fuzzy inference system acts as secondary controller with the objective to augment the performance of the primary controller.
- **Model-based fuzzy logic control**; through fuzzy system identification, a model of the process is estimated and used to control it. A specific type of model-based FLC is *fuzzy model reference learning control*, which will be further discussed in Section 2.4.1.

First, direct FLC uses a fuzzy inference system to directly map an input to a control output for the system. A schematic block diagram of a direct closed-loop FLC and its sub-elements is shown in Figure 2.2. A reference signal is fed through the *fuzzification* block, which transforms the (crisp) signal value into a fuzzy set by determining its degree of membership to the antecedents. Next, the output fuzzy set is obtained by passing the input through the inference mechanism and the rule base. Finally, the output can be defuzzified before it is passed as control command to the system.

Mamdani (linguistic) controllers are commonly used in such a control architecture, with either fuzzy or singleton consequent propositions. The research applications of Mamdani controllers span many

different topics, with early applications being mainly in industrial process control [11], such as temperature regulation [12] or suppressing the swing of overhead cranes [13]. More recently in an aerospace application, a Mamdani controller is used in [14] to directly control the power output of a hybrid fuel cell/battery systems for UAVs. TS controllers can also be used as direct FLC approaches, and resemble gain scheduling approaches. In such applications, the fuzzy system consists of several linear controllers. The controllers are all activated in different magnitudes based on the degree of fulfilment of the input to the rules. In [15], TS gain scheduling is used to control servo-pneumatic actuators in their non-linear operating regions.

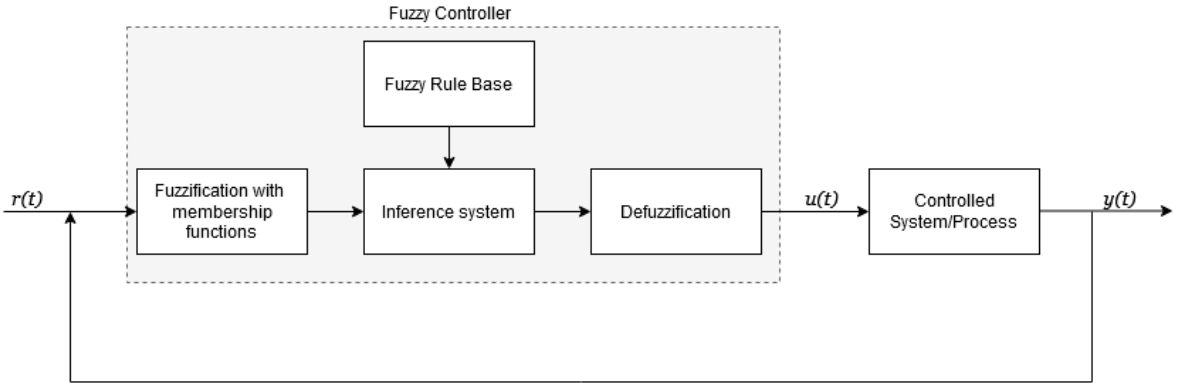

Figure 2.2: Direct FLC architecture

Next, supervisory (indirect) FLC is considered. The FLC itself still has the same structure and elements as in Figure 2.2, however it now acts as a secondary controller which tunes the parameters of the main controller. The supervisory FLC can take information from the process, primary controller, and/or from external sources as input for its inference mechanism. The output of the supervisory controller then adjusts the control behaviour of the primary controller. The advantage of supervisory FLC, is that it can simply be added to existing controllers as opposed to needing to design an entirely new controller. For example, a simple linear controller can be fitted with a supervisory FLC to increase its performance in non-linear regions of the system. Such a system is used in [16], in which the linear controller of a cement mixer is improved with a supervisory FLC. In addition to direct control, a direct TS controller with linear consequent functions, such as in [15], can also be seen as a simple form of supervisory control, since it essentially represents an adaptive combination of linear controllers.

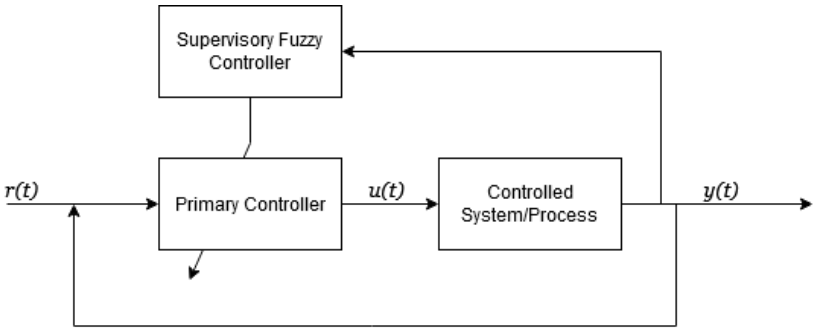

Figure 2.3: Supervisory FLC architecture

2.2. Prioritised Planning

Another heuristic control approach is prioritised planning. Prioritised planning was first introduced by Erdman and Lozano-Perez in their research in [17] in 1987. This control approach is specifically designed for motion planning of agents in a multi-agent system. In prioritised planning, each agent is assigned a unique priority, after which the algorithm solves a shortest-path trajectory problem sequentially starting from the highest priority agent. In pseudo-code, the algorithm of prioritised planning is

shown in Algorithm 2.

Algorithm 2: Classical Prioritised Planning [18]

Algorithm *PP*:
$\Delta \leftarrow \emptyset$;
for $i \leftarrow 1 \dots n$ **do**
  $\pi_i \leftarrow$ `Best-traj`$(\mathcal{C}, \Delta)$;
  if $\pi_i = \emptyset$ **then**
    report failure and terminate
  end
  $\Delta \leftarrow \Delta \cup R_i^{\Delta}(\pi_i)$;
end

Function *Best-traj*$(\mathcal{C}', \Delta)$:
return optimal satisfying trajectory for agent i in $\mathcal{C}'$ that avoids regions Δ if it exists, otherwise return $\emptyset$

In the algorithm, Δ resembles the dynamic occupation space of the planned agents, which is naturally initiated as an empty set. For each of the n agents in the system, a policy π_i is determined based on the optimal possible trajectory at that stage. This is dependent on which spaces are already occupied at which time instance, contained in Δ, and on the static work space $\mathcal{C}$, which resembles stationary obstacles and the bounds of the system. The trajectory of the highest priority agent only has to avoid collisions with the static work space, whereas lower-priority agents must also avoid collisions with the trajectories of previously planned agents. In dense environments, it may occur that there is no possible trajectory anymore for the i^{th} agent in the priority sequence (i.e. $\pi_i = \emptyset$). When the algorithm fails to find a feasible trajectory for an agent, the planning algorithm is terminated and possibly a change in priority sequence is needed.

2.2.1. Computational Effort

There are a number of important characteristics of prioritised planning compared to other trajectory planning approaches. Since the trajectory of each agent is planned independently, prioritised planning is a fully decoupled (and thus uncoordinated) approach. This increases the speed with which the sequential trajectory planning can be performed, making it generally fast enough for real-time applications. In [19] Van den Berg and Overmars compare the computational time of their own prioritised planning approach with a coordinated, optimisation-based algorithm. In their results, they show that the prioritised planning algorithm is much faster and, with their hardware, always provides a feasible solution within 1 second. On the other hand, the optimisation-based approach usually required a computational time a factor 100-500 times higher. Furthermore, this approach became computationally intractable for systems with more than 3 agents. Albeit to a lesser extent, the prioritised planning approach also suffered under an increase in agents, which would cause the computational time to increase exponentially, as shown in Figure 2.4.

To further reduce the computational time, Velagapudi et al. in [20] investigate the application of various distributed controller architectures for prioritised planning. Similar to the structures described in Section 2.3.2, a distributed control structure divides the central path planning problem over the individual agents in the system, resulting in each agent simultaneously determining its own path. In such a system, each agent knows its own static priority, but must communicate with other agents to receive and transmit path and priority information. After this information exchange, a new iteration of path planning is performed by each agent, now taking into account the trajectories of higher-priority agents. Velagapudi et al. in [20] prove that, in a system with n agents, such a distributed approach yields the same solution as the centralised approach within $n+1$ iterations. Furthermore, they propose a number of different communication strategies for the agents, which vary in what information is sent to which agents. Compared to centralised (classical) prioritised planning, the distributed structures discussed in [20] are able to reduce the computational time by a factor 2. A shortcoming of this distributed method, is that the iteration cycles of each agent are globally synchronised. This typically becomes a problem for non-homogeneous systems, since synchronisation forces faster computing agents to wait for slower

agents, thus inefficiently distributing the computational load.

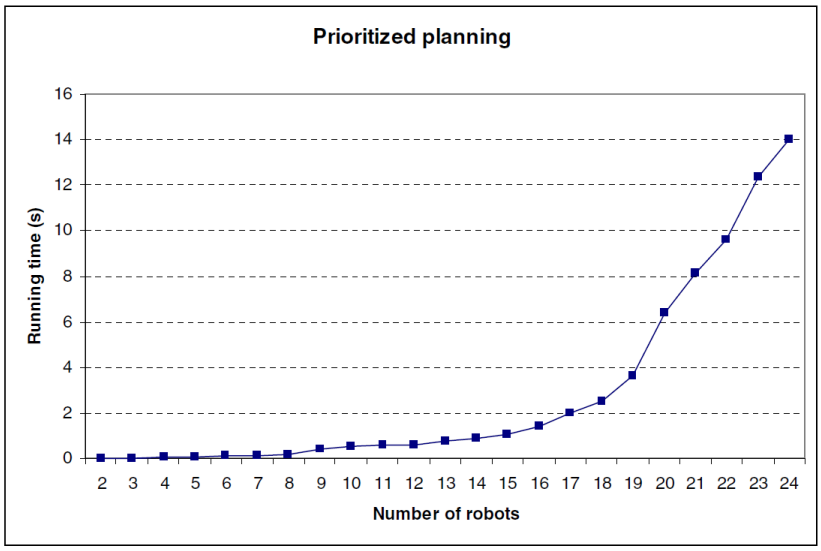

Figure 2.4: Computational time of a prioritised planning algorithm as function of the number of agents in the system, by Van der Berg and Overmars [19]

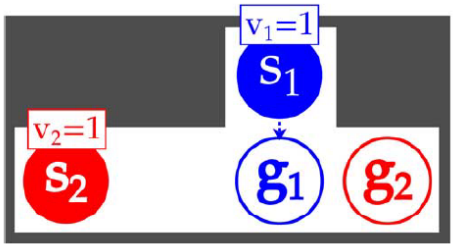

Figure 2.5: A conflicting priority sequence in prioritised planning leading to no feasible solutions, taken from [18]

2.2.2. Incompleteness

A major drawback of prioritised planning approaches is their *incompleteness*. By definition, an incomplete algorithm is a method that can fail to produce a solution to a problem that is in fact solvable. When looking at Algorithm 2, for prioritised planning this would mean the following. At some point in the priority sequence, agent i will have no possible trajectory to follow, despite the fact that the problem could have been solved with, for instance, a different priority sequence. One example of this can be seen in Figure 2.5. In this scenario, agent 1 starting at s_1 has first priority to receive a trajectory to its destination g_2. However, this leaves no feasible solution for agent 2 to reach its own goal of s_2, since all possible trajectories will conflict with the higher-priority trajectory of agent 1. Clearly, there is a feasible solution to the path planning problem of this system, since switching the priorities of agents 1 and 2 would ensure that agent 1 will simply wait before agent 2 has passed before moving to its own destination.

There are a number of other conflict types that can occur in prioritised planning, making classical prioritised planning an incomplete algorithm. Hence, there is an apparent need for a formal understanding under which conditions prioritised planning approaches are guaranteed to yield a solution. To address this problem, Čáp et al. formulate an extended approach in [18] called *revised prioritised planning* (RPP). They prove that RPP is guaranteed to yield a solution if 1) there exists a solution at all, 2) each agent avoids the starting position of lower-priority agents, and 3) each agent avoids the goal position of higher-priority agents. By imposing these additional constraints on the planning approach, Čáp et al. formally prove that RPP is a complete algorithm. One drawback is that agents controlled

by RPP will preemptively avoid certain starting and goal regions in the environment, which generally increases the length of their trajectory.

2.3. Model-Predictive Control

The next control approach covered in this literature study is model-predictive control (MPC). MPC is a collective name for a family of control methods which use a model of the system to find a next-step control action, based on predicting an entire sequence of optimal control actions given the current measured state of the system. Clarke et al. in [21] describe an early form of MPC (referred to as *generalised predictive control*) and show it can outperform a PID controller when applied to a simple plant model. Furthermore, Rawlings et al. in [22] present an extensive foundation for the theory and design of MPC. Moreover, the 2^{nd} edition of the book has added various advances in the field of modern MPC, thus making it a fairly up-to-date source. In addition to the fundamentals of MPC, Rawlings et al. also discuss decentralised, distributed, and cooperative MPC of either coupled or uncoupled systems. Additionally, a comprehensive survey of such algorithm architectures for MPC is given by Scattolini in [23]. These sources will form the main information sources for the review on MPC presented in this section.

First, Section 2.3.1 discusses the fundamental theory of MPC along with its most important characteristics related to control applications. Next, a brief overview is given of decentralised and distributed MPC architectures in Section 2.3.2. This is followed by Section 2.3.3, which discusses an extension to deterministic MPC which makes it possible to incorporate stochastic processes in the control strategy of an MPC-based approach.

2.3.1. Fundamentals of MPC

First of all, a theoretical basis for MPC is reviewed. In Equation (2.7), the standard formulation is given for modelling the dynamic progression of a deterministic, time-invariant system:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ y_k &= h(x_k, u_k) \end{aligned} \tag{2.7}$$

In Equation (2.7), x_{k+1} is the one-step-ahead value of the state vector, x_k is the current state vector, u_k is the current control input vector, and y_k is the current system output. Such a system forms the basis for all deterministic MPC-based approaches discussed in this section (either linear or non-linear). At its basis, all MPC methods consist of three main steps :

1. A process model is used to predict the process outputs or states at future time steps (in discrete time) for a given *prediction horizon*.
2. An optimal sequence of control inputs is determined over a given *control horizon* by minimising a cost function.
3. Of this control sequence, only the first action is implemented. The horizon is moved forward for one control time step and the optimisation is performed again.

In Figure 2.6, these basic principles of MPC are shown. At present time step k the process output $\hat{y}$ is predicted over the prediction horizon H_p. The optimal control sequence is then computed over control horizon H_c, and its first action is implemented. To find the optimal control sequence, a quadratic cost function is commonly used, such as the one shown in Equation (2.8):

$$J = \sum_{i=1}^{H_p} ||r_{k+i} - \hat{y}_{k+1}||_{P_i}^2 + \sum_{i=1}^{H_c} ||\Delta u_{k+i-1}||_{Q_i}^2 \tag{2.8}$$

The cost function consists of two main parts. The first term is the cumulative squared error of the process reference signal. This value represents how well the process is able to follow the reference, and penalises any mismatch between the two. The second term represents the penalty imposed on the control effort. In most processes, the control action requires some form of power, energy, or 'resource' in general, which is preferably kept at as low as possible. Both the signal error and control effort

are weighted with weight matrices P_i and Q_i, respectively. The values of both matrices determine the importance of the two elements with respect to each other. By finding the argument for which the cost function is minimised, the optimal control sequence is determined. At each control time step k, this can be denoted as follows:

$$J^*_{H_p}(x_t) = \min_{\pi} J_k(x_\kappa, \pi) \tag{2.9}$$

subject to:

$$x_{\kappa+1} = f(x_\kappa, u_\kappa) \tag{2.10}$$

$$y_\kappa = h(x_\kappa, u_\kappa) \tag{2.11}$$

$$\pi_\kappa \in \mathbb{U} \tag{2.12}$$

In this notation, J^* represents the cumulative optimal cost function value within the prediction window, and π is the control sequence of length H_p. Furthermore, constraints 2.10 and 2.11 indicate that the state and output vectors adhere to the system model dynamics, and constraint 2.12 states that the control sequence should belong to the total control space. The rationale of MPC is to convert a control problem to a finite horizon optimisation problem of this form. This problem can be solved in a number of ways. One such way is iteratively with the Nelder-Mead algorithm [24], although this is a computationally intensive method and thus usually only applicable to small systems. Another optimisation approach is with dynamic programming (DP) branch-and-bound techniques. Such methods can quickly eliminate large branches of possible control sequences, which can speed up the optimisation problem. However, with increasing variables, the number of branches increase exponentially. Furthermore, there exist various optimisation techniques which first linearise the process either in the current control time step or in multiple control time steps. Then, the optimal solution can easily be found with an ordinary least squares (OLS) estimator, which greatly reduces the computational effort required. This comes at the cost of reducing the model fidelity in strongly non-linear regions of the system, and thus the controller performance.

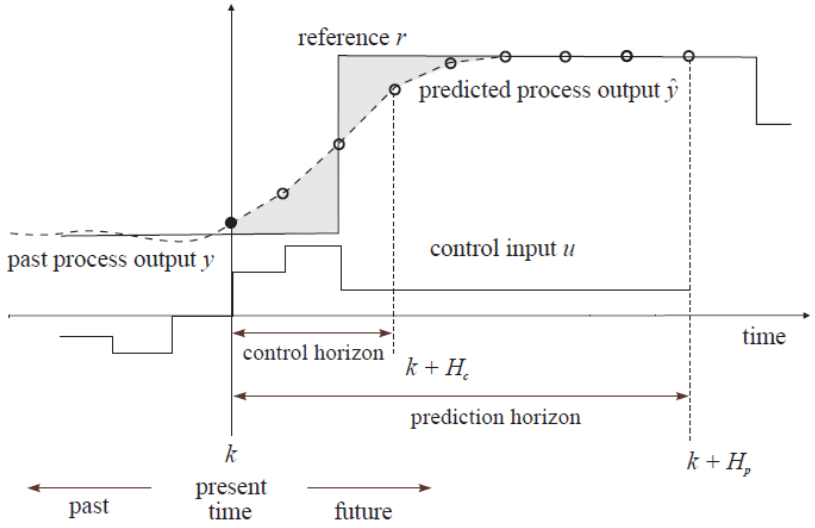

Figure 2.6: Fundamental MPC scheme [25]

From the different optimisation methods, it is apparent that a key characteristic of MPC methods is their computationally heavy nature. Since each optimisation calculation must be performed in real-time for each time step, early applications of MPC systems were only suitable for slow industrial processes [26]. In more recent research, methods have been developed to reduce or mitigate this computational load by developing faster algorithms such as Wang and Boyd propose in [27]. Another strategy is by means of distributed or decentralised MPC architectures, which will be discussed in more detail in Section 2.3.2. An advantage, however, is that MPC is an optimisation-based approach. This results

in a locally optimal solution within the finite prediction horizon, although the solution is not guaranteed to be a global optimum. Furthermore, MPC-approaches incorporate both current and future states in their optimisation strategy. This makes them more likely to converge to a global optimum. Another advantage of MPC, it is very straight-forward to impose constraints on the state and control input of the system. This characteristic makes it possible to incorporate, for instance, limits on actuators or no-go zones and physical obstacles in a simulation/test environment. Since MPC is a model-based control approach, its accuracy, performance, and stability characteristics are heavily dependent on the fidelity of the model used by the controller. It is thus necessary to have a sufficiently accurate model to successfully apply MPC to a system.

2.3.2. Decentralised & Distributed MPC

With the basic theory and characteristics of MPC systems discussed, it is possible to review more specialised MPC-based control approaches. This section starts with a survey on distributed and decentralised MPC architectures. Both architectures are used for large-scale systems, for which it may be more convenient to divide them into smaller and simpler subsystems. Whether a decentralised or distributed MPC structure is required, depends on how interlinked the different subsystems are.

First of all, decentralised MPC requires no strict communication between the individual subsystems, and can thus be applied to decoupled or loosely coupled systems. Compared to centralised control, which optimises the global control problem for all decision variables, the local controllers in decentralised control only optimise their own local variables, without taking into account the control inputs of other controllers. The communication between the local controllers is shown in Figure 2.7.a, for a simple system consisting of 2 subsystems. This yields a simpler implementation of the optimisation problem, however it may reduce the performance of the system if it is more tightly coupled than assumed when decentralising the problem [22]. For further reference, a pioneering survey on decentralised control is presented by Sandell et al. in [28], while a more recent review is given by Šiljak in [29].

In a distributed MPC system, it is assumed that some information is exchanged between the local controller in such a way that both have some awareness of the behaviour of the others, as can be seen in Figure 2.7.b. This MPC structure assumes that the individual subsystems are strongly coupled, meaning that any optimisation performed on one controller will influence the behaviour of one or more other controllers. In terms of coupling connectivity between the subsystems, a system can be either *fully connected* or *partially connected*. In a fully connected system, information from any local controller is transmitted to and received by all other local controllers. On the other hand, a subsystem in a partially connected system will transmit (and receive) this information only to a subset of the other local controllers. Such a structure is convenient to use for large systems with many loosely coupled subsystems and only several strongly coupled links, since it can greatly reduce complexity and computational effort. Furthermore, this information exchange can occur at different rates. In so-called *iterative algorithms* each local controller receives information multiple times per discrete time step, as opposed to only once per time step in *non-iterative algorithms*. Iterative approaches are more computationally expensive, however they are more likely to converge to a (global) optimum due to the availability of more information. The last sub-class of distributed MPC relates to the optimisation behaviour of the local controllers. In some cases, the local controllers all work together to optimise a central, global cost function, which is referred to as *cooperative* control. This can, for instance, be relevant for large chemical plants with multiple process steps, in which the behaviour at each step influences the performance of the next steps. When a local controller only optimises its own local cost function (or even actively reduce the performance of other controllers), one speaks of *independent* or *non-cooperative* control. An example of such systems could be in search robotics, with targets that are actively trying to avoid detection.

These various classifications of decentralised and distributed MPC algorithms are important for organising and categorising research, hence the most important features are summarised in Table 2.1.

2.3.3. Stochastic MPC

In practice, a controlled process is subject to some degree of uncertainty. In Section 2.1, this uncertainty took the form of fuzzy ambiguity and vagueness, which is generally referred to as non-stochastic uncertainty. On the other hand, stochastic uncertainty encapsulates all randomness due to process or measurement noise which can be expressed in mathematical terms. Stochastic uncertainty can thus also be dubbed the 'known unknowns' of a system, since an understanding of the system has made it

Table 2.1: Classification of decentralised and distributed MPC types

Information Exchange/Use	Exchange Frequency	Optimisation Strategy
Partially connected Not all controllers exchange/use information from each other	**Iterative** Multiple updates per time step	**Cooperative** All local controllers optimise a global cost function
Fully connected All controllers transmit, receive, and use information from all others	**Non-iterative** One update per time step	**Non-cooperative/independent** All local controllers optimise a local cost function
Fully decoupled No use of information from other controllers (*decentralised*)		

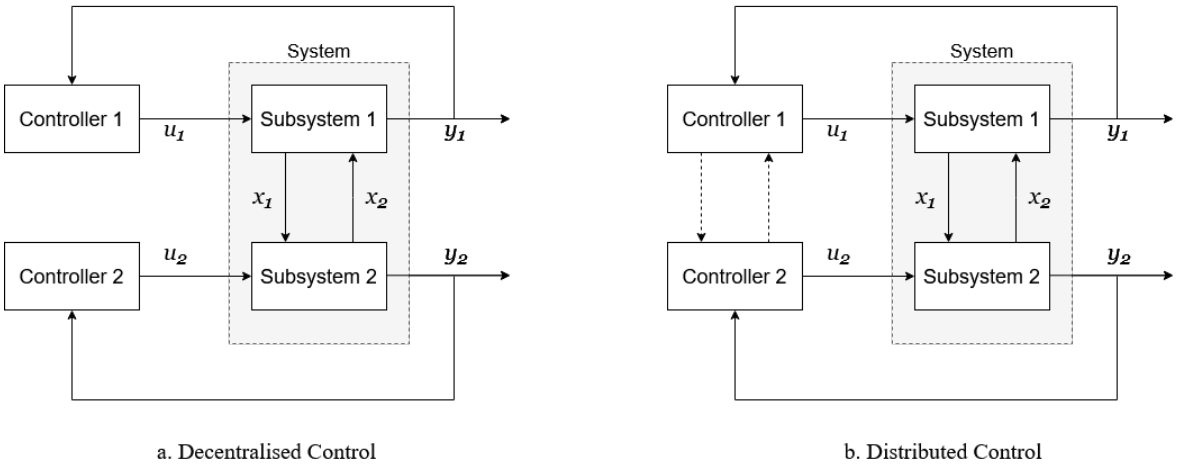

Figure 2.7: Block diagrams of both decentralised and distributed control

possible to account for a probabilistic phenomenon.

To incorporate stochastic processes in a control strategy, stochastic MPC (SMPC) can be used. It must be noted that, to a certain degree, MPC can already deal with system uncertainties, due to its feedback-based nature and repeated receding-horizon optimisation at each time step. However, when such uncertainties become a problem for conventional MPC when they are large or when they grow with time. Furthermore, for explicit incorporation of stochastic processes in the control law, a more robust control architecture is needed. In Equation (2.13) the general formulation for modeling the evolution of the dynamics of a stochastic, time-invariant system:

$$\begin{aligned} x_{k+1} &= f(x_k, u_k, v_k) \\ y_k &= h(x_k, u_k, w_k) \end{aligned} \tag{2.13}$$

Contrary to the deterministic formulation in Equation (2.7), a stochastic system includes disturbance terms v_k and w_k, which can be due to measurement or system noise. Both v_k and w_k have their own (known) probability distributions P_v and P_w, respectively. This results in the cost function Equation (2.8) now also being subject to stochastic terms. To optimise this problem, the cost function is usually subject to chance constraints, which can for instance represent certainty thresholds which the control sequence must satisfy. The new optimal cost value is now denoted as follows:

$$E[J^*_{H_p}(x_t)] = E[\min_{\pi} J_N(x_k, \pi, v_k, w_k)] \tag{2.14}$$

subject to:

$$x_{k+1} = f(x_k, u_k) \tag{2.15}$$

$$y_k = h(x_k, u_k) \tag{2.16}$$

$$\pi_k \in \mathbb{U} \tag{2.17}$$

$$P_{x_k}\{\hat{y}_k = y_k\} \geq \beta_j \qquad \forall j = 1, \dots s \tag{2.18}$$

When compared to the optimisation problem for deterministic MPC, this form includes the additional chance constraint 2.18, which denotes that the probability thresholds β_j should be satisfied. This allows for both fulfilling the control objectives, as well as guaranteeing that all s chance constraints due to uncertainty are met.

For an extensive survey of SMPC, its history, sub-types, and open challenges, the reader is referred to [30]. In this review, Mesbah states that the various approaches for SMPC can be mainly divided into approaches for either linear or non-linear systems. Within both, he argues that there is no clear way to classify all different approaches, although methods can be roughly grouped based on different features in the algorithms. All SMPC approaches do share a number of central challenges, in addition to the general challenges of MPC described in Section 2.3.1. First, the chance constraints are non-convex and computationally intractable. Thus incorporating these constraints in the optimisation problem, makes the problem non-convex and computationally intractable as well. Furthermore, the process of propagating uncertainty through the dynamics of a system adds to the (already high) computational complexity of SMPC approaches, particularly for non-linear systems. Apart from these challenges, a remaining open challenge is establishing proofs for theoretical properties of SMPC-based controllers, such as stability and convergence.

2.4. Combined Controllers

This section serves as a brief review of various combinations or hierarchical adaptations of the previously discussed control approaches.

2.4.1. Fuzzy Model Reference Learning Control

The first integrated controller type that will be discussed, is fuzzy model reference learning control (FMRLC). This control type was first developed by Layne and Passino in [31], in which they combine a FLC with a learning mechanism, as shown in Figure 2.8.

The fuzzy system forms the primary controller, thus making this a form of direct FLC. Additionally, a supervisory learning mechanism is added in the form of model-reference adaptive control (MRAC). In this control structure, a reference model determines a desired model output y_m, which represents a set of pre-defined design goals. This model output is compared with the measured output of the actual controlled process y. Any mismatch between the two values represents an error in the FLC, which the learning mechanism will try to cancel. This learning mechanism will tune the consequent parameters in the FLC such that the closed-loop behaviour of the process matches the reference model. The learning process consists of two parts. First, the *fuzzy inverse model* is used to map the necessary changes which need to be made in the output signal such that the model error is reduced to 0. Secondly, the *knowledge-base modifier* maps these output changes to required modifications for the fuzzy rule-base.

The main advantage of such a learning control approach is that the performance of the primary controller can be improved both autonomously and iteratively. Compare this to conventional FLC, in which the fuzzy parameters are determined beforehand in a very subjective and qualitative fashion, without concrete justification. A controller that can train itself to tune these parameters can greatly increase its control performance. This way, the strengths of a FLC are kept, while mitigating the subjective nature of the fuzzy rule-base parameters. However, it must be noted that the adaptive learning mechanism still contains learning parameters of which the values are still chosen in a relatively arbitrary fashion, without strong theoretical motivation. Furthermore, it is still to be determined if a FLC in FMRLC can achieve optimal control in non-linear systems [31].

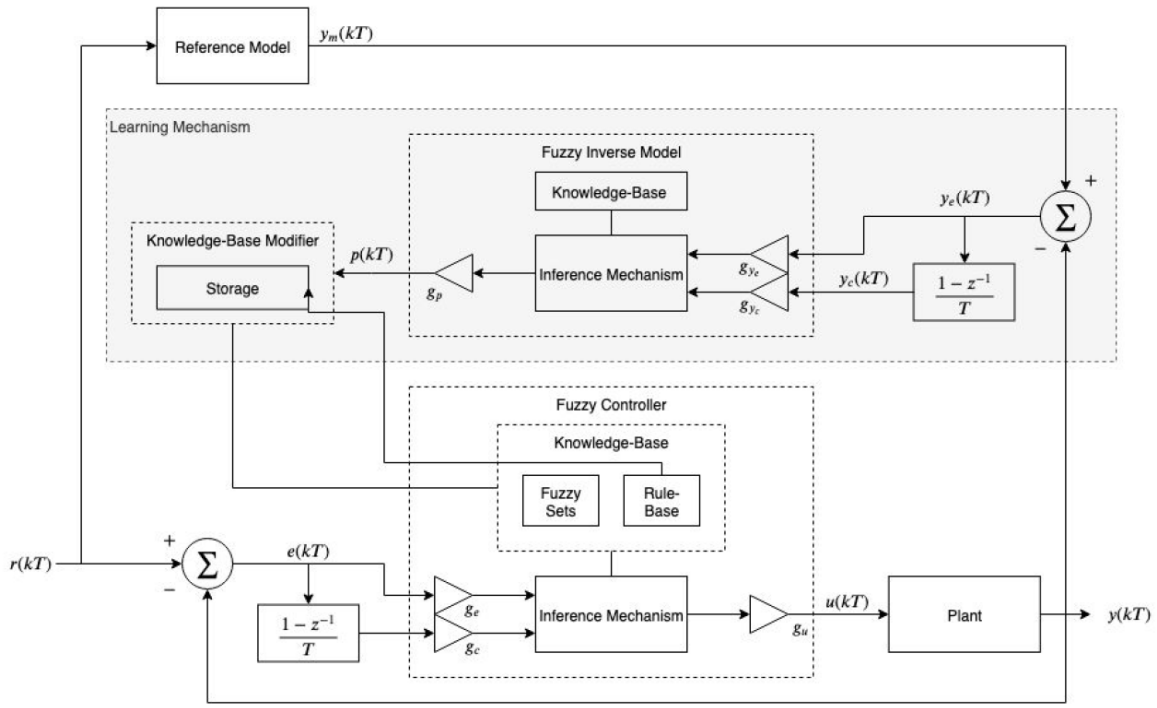

Figure 2.8: Block diagram of fuzzy model reference learning control, from [31]

2.4.2. Feedback Linearisation with Model-Predictive Control

Another type of non-linear control approach, is by means of feedback linearisation (FBL). In FBL, a non-linear feedback method is combined with the system, such that the to-be-controlled system is effectively reduced to a linear system. This way, a simpler linear control law can be applied to the system. However, conventional FBL techniques cannot incorporate constraints on the input and output space. This can be problematic, since FBL approaches can often result in output commands are beyond the bounds of the output space, causes scenarios such as actuator saturation [32].

From this rationale, FBL techniques were combined with MPC approaches, with the first examples coming from the chemical process industry [33]. A simple block diagram of such a control approach is shown in Figure 2.9. The combination of FBL and MPC is symbiotic. Firstly, the MPC approach allows for easy incorporation of input and output constraints in the control scheme. Secondly, the feedback controller enables the use of a linear MPC approach. This greatly reduces the computational load of the algorithm, as opposed to the non-linear MPC approach which would alternatively be required. In [32], Van Soest et al. show that an MPC and FBL can be an effective control method for flight path following. Moreover, such a combination outperforms that of FBL with a PID controller, which was found to show stronger oscillations on the control input. A drawback of FBL combined with MPC, is that FBL is heavily dependent on the fidelity of linearisation feedback. Any model error that is introduced in this block, adds to the total system error (in addition to the possible model error introduced by the MPC block). Furthermore, an open challenge remains to perform a formal stability analysis of such a combined control approach.

2.5. Summary

This section reviews the theoretical basis and characteristics of various control approaches. To summarise this study, Table 2.2 shows the main advantages and disadvantages of each control approach.

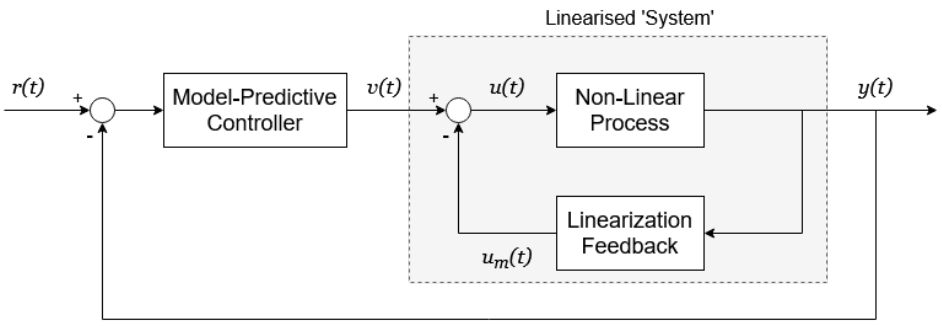

Figure 2.9: Block diagram of a feedback linearisation controller combined with a model-predictive controller

Table 2.2: Summary of all discussed controller characteristics

Controller	Advantages	Disadvantages	Notes
Fuzzy Logic Control	• Transparent logic structure • Simple integration of expert knowledge • Computationally efficient non-linear approximators	• Non-optimal • Performance dependent on quality of rule-base parameters	
Model-Predictive Control	• Powerful estimator for sub-optimum • Simple integration of system constraints • Incorporation of present and future states	• High computational load • Performance dependent on quality of system model	Mitigate computational load with distributed structures
Prioritised Planning	• Fast, efficient approach • Simple algorithm structure	• Incomplete approach	Incompleteness can be mitigated with RPP [18]
Fuzzy Model Reference Learning Control	• General advantages of FLC • Self-tuning of fuzzy parameters	• Ad-hoc selection of learning parameters	Not proven to be optimal
Feedback Linearisation with MPC	• General advantages of MPC • Non-linear system is reduced to linear control problem	• Process model errors contribute in both MPC and FBL block • Possible stability issues	

3

Search-And-Rescue Systems

This chapter reviews current research into SARS. By evaluating and comparing this research, it is possible to identify the state-of-the-art and open challenges of this field. This provides the basis for the motivation of this proposed thesis, as well as forms a theoretical framework for a number of design choices in the research.

In order to break down the literature available on SAR missions, a number of important research aspects are identified. These different elements form the research focus points, constraints, and assumptions of the literature. A comparative discussion between SARS articles is given for each research aspect. First, Section 3.1 addresses the disaster environment in which the SARS is deployed. Another major aspect of any SARS is the hardware that it uses for its agents, which is discussed in Section 3.2. This elaborates on the type and number of agents in the SARS, which is also referred to as *fleet composition* in this literature review. Furthermore, Section 3.3 discusses the required sensing capabilities of the individual agents in the system. This section also briefly addresses the problem of simultaneous localisation and mapping (SLAM), which can strongly influence the efficiency of the agents' search behaviour. Next, Section 3.4 elaborates on the different aspects and considerations of SAR mission planning, followed by a review of the different system uncertainties of SAR applications in Section 3.5. Finally, Section 3.6 summarises the key aspects of each discussed article.

3.1. Environment

One of the most important characteristics of any SARS, is the environment or disaster setting in which the system is deployed. Dependent on the specific scenario, the SARS may have different objectives, or may need to account for different system dynamics.

A distinction can be made between indoor and outdoor SAR missions. A key characteristic of indoor SAR settings, is that the environment is more cluttered than large-scale and sparsely occupied outdoor environment. This may, for instance, pose challenges for the visual recognition and object avoidance capabilities of the SARS agents. As a consequence of these smaller-scale environments, the SARS agents must also become smaller in size in order to manoeuvre efficiently through the environment. This limits the size of the agents, and thus also limits their computational and sensory capabilities [34]. Examples of indoor SARS applications can be found in [5], [34], and [35].

For outdoor SAR missions, the operational environment can vary strongly from indoor settings. Since outdoor SARS are generally applied to large-scale environments, their agents can be larger and more powerful in terms of speed, range, computational power, and sensory capabilities. Furthermore, the obstacles and dynamic elements within the environments are more sparsely distributed, as opposed to a more cluttered indoor environment. For this literature review, three different scenarios can be distinguished within outdoor SAR missions: urban, wilderness, or marine. The terminology for each of these settings is USAR, WiSAR, and ASR, respectively. The type of outdoor setting influences the SAR mission profile. For example, USAR missions generally involve searching for an unknown number of stationary victims trapped within (collapsed) structures, of which the location may be known with relative certainty. In contrast, ASR or WiSAR missions are usually characterised by a fixed number of moving search targets, such as drifting life boats, or mobile humans [36]. Furthermore, the marine environment

of ASR missions are almost always devoid of environmental elements such as buildings, debris, or natural landmarks. Such a sparsely occupied environment can influence the used mission planning methods, such as trajectory planning for the agent(s). Applications of USAR systems can be found in [37], [38], and [39]. WiSAR missions have been widely researched, and successful applications of such rescue systems are presented in [40], [41], [42], and [43]. Finally, in [44] De Alcantara et al. apply a SARS to an ASR scenario. In their research, De Alcantara et al. explicitly incorporate and analyse the effect of wind in their mission planning approach. More generally, this research shows the large influence that meteorological effects can have on the performance of any outdoor SARS.

3.2. Fleet Composition

In terms of fleet composition, this literature review is mainly focused on multi-agents SARS research. Also referred to as agent equipment, the term fleet composition is used to classify the hardware and software specifications of different robots in a multi-agent SARS.

Physical differences are the most obvious type of variation between robots in SARS applications. First of all, and strongly dependent on the environment in which the agent is deployed, is the distinction between land-based, aerial, or sea-based robots. This review will mostly cover aerial robots, more commonly referred to as UAVs, although a number of relevant studies into land-based SARS agents will be mentioned. For instance, Casper and Murphy in [5] are among the first to apply land-based crawlers to a combined indoor-USAR disaster setting. These robots were chosen for their ability to fit and safely manoeuvre within rubble voids with diameters as small as 0.5m. Further applications of land-based robots in SARS can be found in [35], [38], [45], and [46]. Contrary to robots such as crawlers, which are limited to land-based operations, robots such as rotary or fixed-wing UAVs are exclusively airborne. Goodrich et al. in [42] and [43] identify a number of key benefits of airborne robots (and specifically fixed-wing UAVs) for WiSAR applications. The main advantage is the ability of UAVs to quickly cover large search areas of difficult terrain, which may be challenging to navigate or completely inaccessible for land-based operations. Similarly, De Alcantara et al. in [44] investigate the use of fixed-wing UAVs for ASR missions, since they are able to traverse a marine/coastal environment while providing a 'birds-eye view' of the search area. With rapid technological advances in airborne drones of the past decade, research into UAVs in SARS applications has also increased. Further examples of such UAV SARS research can be found in [34], [37], [38], [40], [41], and [47]. The cases of both aerial and land-based robots show the influence of the disaster environment on the robot type.

Furthermore, when analysing multi-agent SARS applications, one can make an extra distinction between homogeneous and non-homogeneous (also referred to as heterogeneous) fleets. In a homogeneous fleet, all agents are identical in terms of their performance capabilities, functionality, and equipped sensors. This can add a degree of flexibility and robustness to the SARS through redundancy, since all agents are equally well suited to execute any of the mission tasks. Hence, any agent is able to replace any other agent in case of a failure, as opposed to one agent having to replace an agent which is specialised for a certain type of task. However, this research focuses on the application of non-homogeneous agents. In a non-homogeneous multi-agent SARS, different agents have different capabilities or hardware specifications, and as a result may be more specialised or efficient for different tasks within the SAR mission. In [38] Beck et al. split their SARS agents into both search robots and rescue robots. The reasoning is that robots tasked with searching for victims will generally require high speed and an advanced sensor base for victim detection, whereas rescue robots will need manoeuvrability and actuators to provide aid to found victims. Furthermore, Tanzi et al. in [47] propose a SARS fleet architecture of 3 different UAV types: vertical axis drones for indoor manoeuvrability, fixed-wing drones for fast area coverage, and blimps as communication infrastructure. Tanzi et al. argue that a SARS can be more efficient if different drone capabilities are exploited in the mission planning strategy, similar to the hypothesis of this research. In addition to exploiting the individual capabilities of each agent, the use of non-homogeneous agents can form an economic benefit. For instance, distributing different types of sensors over different agents is financially cheaper than integrating all sensory capabilities on a single agent [48]. In [40], Stecz and Gromada. propose such a SARS with different sensory capabilities distributed over the agents in the system. However, due to lack of a standardised test bed, the developed system could not be compared properly to existing methods. It is thus unclear if the flexible sensor platform increases the victim search efficiency of the SARS.

On a final note, another form of non-homogeneous SARS agents is discussed by Arnold et al. in

[37]. This research focuses on how the victim detection performance is affected by assigning different 'personality types' to the UAV agents in the system. These personality types determine the searching behaviour of an agent. The concept makes the agents in such a SARS non-homogeneous on a behavioural level, instead of from a hardware perspective.

3.3. Agent Sensing & Simultaneous Localisation And Mapping

The highly stochastic and dynamic environments in which a SARS is deployed, pose a major challenge for practical applications of these systems. For example, any prior knowledge that may be fed into the system can be partially or completely invalid in the aftermath of a disaster. One can think of collapsed structures and inaccessible areas due to flooding. Furthermore, a SARS may have to account for moving victims, fire spread, or falling debris during its operations. These challenges lay at the heart of Simultaneous Localisation And Mapping (SLAM), in which an autonomous agent must collect information about the environment while storing the agent's position within it. This way, agents can communicate what they perceive through their sensors and where they have perceived it. For a more comprehensive overview of the theory and history of SLAM, the reader is referred to [49], [50], and [51]. Additional examples of successful applications of real-time mapping of unknown disaster environments are discussed in [35], [46], and [52].

When investigating the sensory and SLAM-solving capabilities of a fleet of SAR drones, it is important to state the purpose of addressing SLAM in a disaster setting: *to collect and relay unknown information about the environment to the mission headquarters, such that improved mission plans can be determined.* Mission planning approaches can be influenced by real-time updates of dynamic changes in the system. More specifically, sensory data gathered by the agents of a system can provide ad-hoc knowledge of such changes. It is thus important to assess what type of information or data is important to the mission planning controller. In [45], Riley and Endsley follow a USAR team and observe the human-robot interaction between the rescue workers and the remote drones they used. In this cooperative system, the human rescue workers are essentially the 'mission planners', manually performing task assignment and path planning for the robotic agents. Since the mission planning module in an autonomous SARS attempts to process input data similar to a human operator, the research of Riley and Endsley can offer a valuable comparison. This first-hand, practical experience gives good insight for assessing what information from the SARS agents is important for the mission planner to receive. In addition to this, in [5] Casper and Murphy provide an extensive post-hoc analysis of the human-robot interaction during the response and relief phase directly after the World Trade Centre attack on 11 September 2001.

Both [5] and [45] strongly emphasise the importance of situational and perceptual awareness of SARS agents. In most SARS research, visual perception forms the primary method for object classification, object avoidance, and risk assessment. This is generally most commonly achieved with optical sensors such as cameras, heat sensors (infra-red), or light detection and ranging (LiDAR) sensors. Visual input from the agents is especially useful if there is a mismatch between the agent's perceptual data and the environment model of the mission planner. Furthermore, acoustic sensors can provide auditory awareness for the agents. Most notably, this can be of use when trying to localise victims based on their calls for help. A SARS application that uses acoustic sensors in combination with optical cameras and heat sensors has been investigated by Ganesan et al. in [53]. Additionally, recent studies have begun investigating the use of wireless sensors to detect passive electronic devices. An early research in this field is performed by Loukas and Timotheou in [54], in which SARS agents are used as mobile wireless routers to detect the cell phones of victims. More recently, Wang et al. in [55] similarly investigate the feasibility of using WiFi-enabled UAVs to detect victims in a SAR environment. In their research, the authors show that it is indeed a viable search method, although it is not as widely used as other sensors yet.

3.4. Mission Planning

As stated in Chapter 1, this research defines mission planning as a twofold problem. First, the problem of assigning a destination or search area to an agent, and secondly, to determine a corresponding path to reach that destination.

First of all, the problem of task assignment arises when there are more tasks than agents in a system. This is more commonly referred to as a multi-robot task assignment (MRTA) problem. Fur-

thermore, this literature study mainly investigates Single-Task robots, Single-Robot tasks, and Time-extended Assignment (ST-SR-TA). This means that every robot can execute one task at a time, every task can be executed by one robot, and each task takes a certain time duration to complete [56]. For this type of task assignment problem, the mission planner must decide which tasks have priority over others at each time step. In a USAR disaster setting, the to-be-searched areas in the search grid form the different tasks to which the mission planning controller must assign a priority score. In [41] San Juan et al. construct a Mamdani FLC approach to give a priority score to each cell in a given search grid, which corresponds to the potential hazard to victims in that cell. This score is based on terrain and emergency factors, such as how frequent the cell is visited by people and how close a SAR team is to that position. In contrast to this, task prioritisation in [44] by De Alcantara et al. takes a far simpler form. Here, assignment of a search area is purely based on which agent is closest to the most desirable location, which is incorporated in an MPC cost function. In a similar fashion, Beck et al. in [38] assign search areas to robots by minimising the cumulative travel time, since it is assumed that the mission tasks are already defined. Apart from prior environment knowledge, real-time updates in the form of in-situ measurements from the agents can form an additional source of information for the task assignment controller. This type of system knowledge is strongly dependent on the sensing capabilities of the agents, which have been discussed in Section 3.3.

Furthermore, the introduction of non-homogeneous agents adds an extra degree of freedom to the task assignment problem. In addition to assigning a priority score to each area in the search grid, the mission planner must also decide which agent is best suited to execute each task. To determine this, the mission planner can incorporate the characteristics of each agent. For instance, a high-speed drone may be the best choice for a distant, high-priority search area with a time constraint on it. In another case, if a building is known to have a compromised structure, a cheaper agent can be assigned to search it, since the consequences of losing this agent are less costly. While performing this literature review, no explicit research has been found into SAR task assignment with non-homogeneous agents.

The second aspect of mission planning is determining the path that each agent follows to reach its sub-goal. Many different approaches have been researched and developed for trajectory planning, with different value criteria that assess which path is chosen. One of the most popular types of path planning methods, are *shortest path planning* approaches. In these approaches, the mission planner always tries to find the trajectory with minimal travel distance or time for the agent. The motive for these approaches is that time and energy are two important cost factors for most systems, thus making it preferable to minimise both. Most of the previously discussed research articles incorporate a shortest path approach. The shortest path problem can be solved in various ways. Heuristic approaches have historically been favoured for their computational efficiency, at the cost of non-optimality. Many heuristic shortest path approaches have been applied to SAR missions, such as reaction-based swarming [37], FLC [41], and ant colony optimisation [54]. However, with advances in computational power and algorithm efficiency, mathematical approaches are becoming more applicable to larger SAR missions. Examples of such approaches in SAR missions are hindsight optimisation (HOP) [38], mixed-integer linear programming (MILP) [40][57], and distributed nonlinear model-predictive control (NMPC) [44].

Another type of path planning methods, are *risk-based path planning* approaches. Generally, such approaches are focused on finding a path that provides a trade-off between being both short and safe. Especially in environments with dynamic, stochastic risks such as in a USAR setting, agents can themselves be at risk of damage or loss. From this rationale, risk-based path planning approaches try to incorporate knowledge of possible environmental hazards and threats into the trajectory of each agent. Applications of risk-based path planning approaches can be found in both [40] and [58].

3.5. Uncertainty

One final way of categorising SARS research, is by what type of disturbances or uncertainties are incorporated in the system. In general, the integration of stochastic processes or disturbances is found to be rarely addressed in current SARS research. However, a reoccurring uncertainty in various research, is the largely unknown location of victims. To incorporate this uncertainty, victim probability distributions can be used as input for the mission planning controller, as is done in [41], [44], and [57]. These probability maps increase the search efficiency of a SARS, since they ensure a 'warm start' when assigning search areas to the agents.

Furthermore, uncertainties such as sensor noise are often assumed to be negligible, if discussed

at all, in most articles reviewed in the previous sections. However, in [46] Chen et al. account for noise in LiDAR sensors by applying a Kalman filter to the measurements. Similarly, Goodrich et al. in [43] address noise in the optical cameras of their UAVs due to low image resolution. This problem was only encountered during field tests of their SARS, and resulted in uncertainty when detecting victims from images. The authors propose to mitigate this noise by using higher resolution sensors. In [53] Ganesan et al. use an image filter for noise elimination of their robot's optical input.

Beck and al. in [38] address uncertainty in the tasks of the SAR mission. As mentioned previously, the authors split the set of all tasks into either search tasks or rescue tasks. In this division, all search tasks are assumed to be predefined in the mission profile, and are thus considered as *known tasks* $\mathbf{T}$. Since a rescue task can only be performed if a victim is found during a search task, it is considered to be an *uncertain task* $\mathcal{T}^+$. In this case, the mission planning controller maximises the expected optimal task assignment solution of the union of both task sets $\mathbf{T} \cup \mathcal{T}^+$. Beck et al. term this approach uncertain multi-robot task assignment (UMRTA).

3.6. Summary & Conclusions

In Table 3.1, a complete breakdown is presented of the utilised literature. The overview forms the basis for establishing the current state-of-the-art of research into SARS, and how this research proposal can add to that body of knowledge. In addition, it is now possible to identify a number of trends throughout the field of SARS research.

First of all, it can be seen that the fleet composition of SARS research from the last decade mostly consists of airborne agents. SARS research with (mostly manual) land-based robots was commonplace before 2010, whereas airborne robots such as fixed-wing UAVs and vertical takeoff rotor craft are emerging as the new dominant type of SARS agents. This can be attributed to the rise of inexpensive, easy to operate UAVs for the consumer market [36]. Combined with the ability of UAVs to traverse difficult or inaccessible terrain, this puts airborne agents at an operational advantage compared to land-based robots. Next, in Chapter 2 it was determined that more computationally demanding control methods, such as optimisation-based approaches, are applied in practice, due to technological advances in processing units. As a result, it can be expected that this trend will also be apparent in the control approaches used in SAR mission planning research. Indeed, it was found that recent research has analysed the applicability of mathematical control approaches to SAR mission planning. However, when reviewing the state-of-the-art of SARS research, it was found that heuristic approaches are still an equally popular choice for this purpose. Both types are viable choices for mission planning, and each have their own merits and drawbacks. Furthermore, today's 'modern' disaster victims almost always carry a cell phones or other wireless device on their person. This has sparked research into the use of sensors that can detect WiFi signals of passive devices. This is proven to effectively extend the sensing capabilities of SAR agents, in addition to conventional sensors such as optical cameras and heat sensors. Finally, uncertainty is frequently left unaddressed in SARS research, with the exception of the uncertain location of victims. Moreover, it is usually only evaluated and accounted for once a real-life test of the SARS has been performed. In such a case, it is found that sensor noise is one of the most common types of uncertainty to be addressed.

Table 3.1: SARS state-of-the-art literature breakdown

Authors & Source	Environment	Fleet Composition	Sensors	Task Assignment	Path Planning	Uncertainty	Notes
Casper and Murphy (2003) [5]	Indoor/USAR	Multi-agent, non-homogeneous, land-based robots	Optical, Acoustic	Manual	Manual		Post-hoc analysis
Sampedro et al. (2019) [34]	Indoor	Single-agent, UAV	Optical	K-means clustering	ROS planner package		
Arnold et al. (2020) [37]	USAR	Multi-agent, non-homogeneous, UAV		Behaviour prioritisation	Reaction-based swarming		
Beck et al. (2016) [38]	USAR	Multi-agent, non-homogeneous, UAV		UMRTA	HOP	Rescue task uncertainty	
Stecz and Gromada (2020) [40]	WiSAR	Multi-agent, non-homogeneous, UAV	Optical		(Risk-based) MILP		Assumes that there are always more agents available than mission tasks
San Juan et al. (2018) [41]	WiSAR	Single/Multi-agent, homogeneous, UAV		Fuzzy prioritisation	FLC (among other approaches used)	Victim probability map	
Goodrich et al. (2008,2010) [42][43]	WiSAR	Single-agent, UAV	Optical	Generalised contour search		In-the-field image noise	Field tested
De Alcantara et al. (2019) [44]	ASR	Multi-agent, homogeneous, UAV		NMPC	NMPC	Victim probability map	
Riley and Endsley (2004) [45]	USAR	Multi-agent, non-homogeneous, land-based robots	Optical	Manual	Manual		Post-hoc analysis
Chen et al. (2017) [46]	Indoor	Single-agent, land-based crawler	Optical (SLAM)			Sensor noise	Field tested
Tanzi et al. (2016) [47]	USAR	Multi-agent, non-homogeneous SARS					Theoretical survey
Schmuck and Chli (2017) [52]	Indoor/Outdoor	Multi-Agent, homogeneous, hand-held camera/UAV	Optical (SLAM)				Field tested
Ganesan et al. (2011) [53]		Multi-agent, homogeneous, land-based robots	Optical, heat, acoustic		Ant colony optimisation		
Loukas and Timotheou (2008) [54]	Indoor	Multi-agent, homogeneous, land-based robots	Wireless		Distributed greedy heuristic		
Wang et al. (2013) [55]	WiSAR	Multi-agent, homogeneous, UAV	Wireless				Feasibility study of using wireless sensors in airborne WiSAR
Berger and Lo (2015) [57]		Multi-agent, homogeneous, UAV			MILP	Victim probability map	
Rudnick-Cohen et al. (2016) [58]	USAR	Single-agent, UAV			(Risk-based) Dijkstra's algorithm		

4

Simulation Design

Finally, a SAR simulator must be constructed to test the performance of the developed SARS. In various literature presented in Chapter 3, it was established that one of the main challenges of developing a new mission planning approach is using an appropriate test platform for the system. For a high-fidelity test environment, it would be preferred to simulate a physical, life-size disaster setting combined with an actual system of UAVs. In practice, such a set-up would prove too costly and time-consuming to develop. Moreover, in the initial stages of any novel system, it is necessary to test first the concept in a virtual simulation. This way, the consequences of any (sub)system failure will be limited to the virtual environment of the simulation. For these reasons, this research will make use of a virtual simulation environment as test bed, in which virtual SARS agents can interact with a disaster environment and search for victims. In the literature, the virtual disaster settings created for validating the proposed solutions are not based on a standard simulation framework. Therefore, making a fair comparison between different solutions is difficult.

This chapter offers a review and discussion of simulation design for disaster settings, which have the purpose of functioning as a test bed for a SARS. In Section 4.1, the different static and dynamic elements that are necessary for a SAR simulator are discussed. This section will take various internationally recognised directives for SAR simulation as a basis. Once the basic objects and elements of a SAR simulator are discussed, Section 4.2 will review various existing simulator environments that may be appropriate for SAR research.

4.1. Simulator Elements

First of all, it is important to review the demands to which a SAR simulator must adhere. For the purpose of this chapter, two main requirements are identified: the simulator must 1) contain appropriate elements which could also be expected in a real-life disaster scenario, and 2) model the characteristics, dynamics, and constraints belonging to each element with a sufficient degree of fidelity. For the first requirement, Robocup-Rescue[1] can be used as internationally recognised reference framework for disaster scenario simulation. Robocup-Rescue was developed after the Great Hanshin earthquake in Japan on January 17 1995. One of the aims of Robocup-Rescue is to provide a standard benchmark for evaluating research into SARS. In [59], founders Kitano et al. summarise the initial proposal for Robocup-Rescue, in which they also list a number of simulation elements for a disaster scenario. These elements are listed below:

- **Building damage**. This simulates the degree of structural damage to houses and other buildings in the environment.
- **Fire spread**. Fire models can simulate how flames spread through the disaster setting.
- **Life-line damage**. This includes simulating damage to infrastructure such as roads, water lines, and power grids.

[1] https://rescuesim.robocup.org/, (Retreived November 2020)

- **Victim modelling**. This models the behaviour and (medical) condition of victims in a disaster setting.
- **Refugee modelling**. This models the behaviour of larger groups of people, such as fleeing the disaster scene.

Over time, Robocup-Rescue has developed more sophisticated open-source models of these simulation elements, which can form the basis for the simulation used to evaluate this research. For reference, the user manual of the Robocup-Rescue simulator[2] is used. Additionally, Robocup-Rescue has added wind effects in its more recent simulation environments. Wind couples strongly with fire spread and trajectory planning of (airborne) agents. The following subsections will briefly discuss the considerations for building damage, fire spread, victim modelling, and wind effects.

4.1.1. Building Damage

In most USAR settings, structural damage to buildings is a common effect of disasters. Furthermore, it can form an obstacle or hazard for both victims and SARS agents during the mission. In the simulator software of Robocup-Rescue, building damage is modelled by attributing properties to all structures in the virtual environment. These properties are mainly the *material* and the *collapse degree* of a building. The materials considered for the simulator are *wood, steel*, and *concrete*. The collapse degree of each building, with increasing severity, are *none, slight, moderate, severe*, and *destroyed*. Buildings with enough structural damage can trap victims inside them. The simulator is initialised by randomly assigning structural damage to buildings with a predefined probability distribution, and does not change the structural properties of buildings during the simulation.

4.1.2. Fire Spread

Gas leaks and electrical short-circuiting have the potential of starting fires during a disaster. In the presence of flammable material and/or strong winds, fires can spread within and between buildings, posing risks for both victims and rescue agents. In Robocup-Rescue, buildings have an additional property termed *fieryness*, which indicates the fire intensity and fire-related damage to the building. With increasing severity, the fire intensity can be *unburnt, heating, burning,* or *inferno*. The fire damage is categorised as either *minor, moderate, severe* or *completely burnt out*. How the fire will spread through the simulator is based on parameters such as the material of a building, the wind intensity and direction, and intensity of the fire.

4.1.3. Victim Modelling

In real-life USAR settings, victims of a disaster may have different levels of movement freedom, either due to injuries, being trapped, or feeling disoriented. In Robocup-Rescue the behaviour of victims is modelled by attributing a number of properties to them. First, a victim's level of *buriedness* will determine what level of movement (if any) is possible, as well as determine the *damage* level of the victim. This second property determines how much *health points* the victim loses per time step in the simulation (i.e. how hurt a victim is). If a victim is reduced to 0 health points, they perish. If the victim is free to move, they will always try to make their way to a safe refuge in the simulator.

4.1.4. Wind Effects

In small UAV missions, wind can quickly exceed more than half of a UAV's airspeed, thus significantly affecting the UAV ground speed. In the case of two agents that are equally distanced from a target, the upwind agent can reach the target in a shorter time and with less required energy. Furthermore, as discussed in Section 4.1.2, the wind direction and intensity can influence how fire sources spread in the simulation environment. Robocup-Rescue incorporates wind in the simulator by setting a time-invariant, uniform wind vector for the entire environment.

4.2. State-Of-The-Art Simulators

As has been shown in the previous sections, Robocup-Rescue has a wide and extensive availability of simulation tools specifically designed for USAR disaster settings and SARS development. The flex-

[2] https://rescuesim.robocup.org/resources/documentation/, (Retreived December 2020)

ible design toolbox allows for custom creation of maps and disaster scenarios. These disaster maps are 2D, and are generally on the scale of a city block or larger. Additionally, the open-source nature and documentation make for a convenient and reasonably supported test platform for SARS research. The historical context which motivated Robocup-Rescue to be developed, shows that its environmental models are primarily focused on modelling earthquake disasters (or disasters with similar consequences). For other natural disasters, such as flooding or hurricanes, the simulator has significantly less supporting tools.

Another popular simulation environment is Gazebo Sim[3]. Gazebo has a wide range of applications, with the ability to accurately and efficiently simulate (populations of) agents in complex indoor and outdoor environments. The sophisticated physics engine and 3D graphics of Gazebo make it a simulator with high-fidelity modelling power. Moreover, Gazebo is a free service with a large, active community. A disadvantage of Gazebo, is that it is not specifically developed for SARS testing. This results in a lack of specific toolboxes and internationally recognised standards that simulators such as Robocup-Rescue do possess. When reviewing whether Gazebo adheres to the requirements of a SAR simulator, it can thus be determined that not all elements of a real-life disaster scenario are present in the software. However, the elements that can be modelled have a relatively high degree of fidelity compared to other simulator environments.

Next, Netlogo[4] is discussed as simulation environment. NetLogo is a 2D modelling environment, used for high-level modelling of multi-agent systems. Similarly to Gazebo, Netlogo is not specifically designed for modelling disaster settings, and thus lacks many of the toolboxes that Robocup-Rescue has. Moreover, its high-level modelling engine makes it a low-fidelity simulator for agent behaviour. Hence, NetLogo adheres poorly to both requirements of SAR simulators stated previously. Nonetheless, Netlogo may be suitable as a simple and fast test platform for (sub)system testing of certain controller architectures.

One final simulator option, is to construct a custom simulator based on the elements and requirements of directives such as Robocup-Rescue. The main advantage of this is that the simulator can be tailored to the needs of the SARS experiments. Therefore, the simulator will contain all relevant elements without any unnecessary features. The risk of this option, is that it will take up too much (time) resources to complete in the time span of this thesis work. This risk can be mitigated by extending the simulator on previous research/development by members of the research group in which this thesis is performed, thus reducing the workload.

[3]http://gazebosim.org/ (Retrieved December 2020)
[4]https://ccl.northwestern.edu/netlogo/ (Retrieved December 2020)

5

Proposed SARS Approach

The purpose of reviewing all literature from the previous chapters, is to establish a theoretical basis of different control approaches, review the state-of-the-art of current SARS, and address the different elements required for an accurate SAR simulator. With this, it is possible to formulate a thesis plan which adds to the body of research related to SARS. To do this, the objective of this thesis work is stated once more:

> To reduce the total number of casualties caused by a disaster, by increasing the victim search efficiency with respect to mission time and area coverage of a multi-agent, non-homogeneous search-and-rescue system of UAVs.

One of the main focuses of this research, is how the non-homogeneous nature of multiple SARS agents can be incorporated in the mission planning approach. The hypothesis is that, by correctly exploiting the individual capabilities of each agent, the victim search efficiency of a SARS can be increased. Hence, the SARS agents in this proposal will have physically different performance characteristics. These performance differences will initially be in terms of speed, range, computational power, battery power/size, and value, with the possibility of adding more parameters at a later stage of the research.

For the mission planning approach, the following architecture is proposed. To mimic the human-like reasoning used in task assignment, it is concluded that FLC is best suited for this aspect of mission planning. An FLC-based task assignment module will be designed to take knowledge from both the environment and agents as inputs, and yields the priority ranking of all search areas with their corresponding agent allocation. This approach has a number of key advantages. First, an FLC-based controller can mimic the inherently human background of task assignment in SAR mission planning; the FLC-based controller will be able to incorporate uncertainty in the form of linguistic inputs (for example, a *high* probability of fire in a search area), and yield crisp singleton outputs. Hence, the task assignment module will have the structure of a direct Mamdani controller. Furthermore, it is possible to initially build up an extensive fuzzy rule-base using the existing knowledge of expert SAR personnel. A form of fuzzy priority assignment was found in [41], although the proposed FLC method of this thesis will incorporate a wider variety of a-priori knowledge and additionally yield the agents' task allocation as an output. It is expected that the integration of expert knowledge in the controller increases its victim search performance, compared to control approaches that cannot incorporate such knowledge. This is due to the hypothesis that the mission planning module will be able to more efficiently assign tasks that have a higher chance of saving victims if it has knowledge of their possible locations and the hazards and obstacles within the environment. The second part of the mission, trajectory planning, will be done by means of a prioritised planning approach. Such an approach is chosen, since its simple architecture is specifically developed for multi-agent motion planning. Furthermore, the priority sequence of the agents will already be a product of the previously discussed fuzzy task assignment controller. This way, the priority computations are efficiently used for both aspects of the mission planning approach. If needed, the computational effort can be mitigated further by distributing the algorithm over the agents, as was found in [20] and [18]. One drawback of prioritised planning, is the possibility of trajectory planning failure due the incompleteness of the algorithm. However, such conflict situations generally occur

in densely packed environments. Since disaster settings are more sparsely occupied compared to the scale of the agents, it is expected that such conflicts will be rare. Nonetheless, the possibility of using the extension of RPP [18] is kept as a mitigation measure. This proposed combination of fuzzy control and prioritised planning for task assignment and trajectory planning is termed *Fuzzy-Logic-Assisted Prioritised Task Assignment and Path Planning* (FLAPTAPP) .

The mission planning approach will only be adaptive in terms of its utilised world model, since it will be updated in real-time with sensory data gathered by the agents. To update the world model of the mission planning approach, the agents will be modelled as UAVs with basic sensing capabilities, such as camera vision. This way, agents can relay information about the environment to the central mission planning module. With this approach, the mission planning controller can adapt its strategy based on new knowledge of the environment, gathered by the agents interacting with it. Furthermore, it is determined that little communication is required between the individual agents in the SARS. Assuming there is a relatively stable communication link between the agents and the mission planning approach, the latter will function as a central distributor of all relevant information about tasks, trajectories, and the disaster environment.

Finally, a custom, 3D simulation environment is developed to test the proposed SARS. It is chosen to build a custom simulator since it is desired to follow a simple, step-by-step approach in the development of the SARS. This involves testing all individual subsystems, before integrating them into a final SARS. It is expected that this is best done in a custom environment with only the most necessary SAR elements included in the simulator. The simulated disaster scenario will be that of an indoor USAR setting, including the modelling of fire spread, victims, and structural damage. Both the modelling of fire spread and victim movement are of a stochastic nature, in order to simulate the uncertainty of their dynamics in a real-life SAR scenario. To ensure the validity of the simulator, these elements and their dynamic models, properties, and characteristics will be based on the internationally recognised standards of Robocup-Rescue. By using their open-source, SAR-specific models, the custom simulator is validated. The main risk of developing a custom simulator for this application, is that it will take up too much time to develop within the time span of this thesis work. To reduce the required workload, previous work related to SAR simulators from the research group, in which this thesis is performed, is incorporated in the development of the simulator environment.

To provide a high-level concept of the proposed SARS, the preliminary architecture of the system is shown in Figure 5.1. With this research proposal for a novel SARS approach, it is possible to reformulate the research question for this thesis work.

> **Can an adaptive FLAPTAPP approach for non-homogeneous agents increase the victim search efficiency of a search-and-rescue system of UAVs in terms of victim search time and area coverage?**

To support this question, a new set of revised sub-questions can be formulated.

- How can a FLAPTAPP approach be applied to mission planning?
  - How is priority assigned to areas in the search grid?
  - How are non-homogeneous agent capabilities integrated in FLC?
  - How are the trajectories determined by the prioritised planning controller?
- What is the communication architecture between the central mission planning controller and the SARS agents.
  - How is mission-related information exchanged between agents and the central mission planning controller?
  - How is the world model of the mission planning controller updated?
  - What protocols must be in place in case of communication disruption?
- How must a simulation environment be designed in order to test the victim search efficiency of a SARS?
  - How are the different static and dynamic elements of the simulator modelled?

– How are the dynamics of the agents modelled?
– How must an experiment be designed for fair comparison between SAR mission planning approaches?

The main contributions of this research are twofold. First and foremost, this thesis work extends on previous research concerning SAR mission planning for non-homogeneous agents. Specifically, this thesis proposal investigates the integration of a-priori knowledge about the agent's capabilities into the task assignment and trajectory planning problem of the SAR mission. During the SAR mission, this system knowledge is expanded with real-time sensory data collected by the agents. The novelty of this research is applying this form of knowledge-based mission planning to a non-homogeneous SARS. This mission planning approach is designed, developed, and evaluated against alternative mission planning methods.

Secondly, a novel control approach is proposed, by combining FLC and prioritised planning in an integrated mission planning controller. This integrated controller architecture is termed FLAPTAPP. While performing this literature study, no previous publications were found which use a control approach similar to that shown in Figure 5.1. This research attempts to design, develop, and evaluate the performance of this control approach against alternative control approaches.

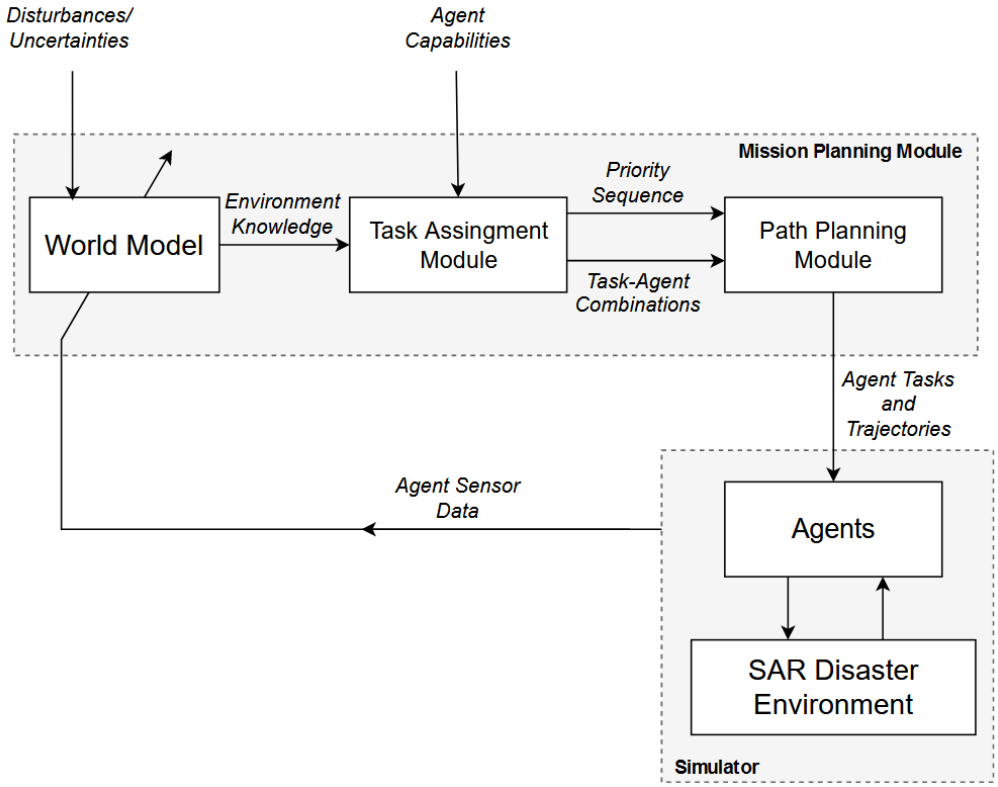

Figure 5.1: Proposed SARS and simulator architecture

6

Conclusion

To increase the victim search efficiency of a robotic search-and-rescue system (SARS) in a disaster setting, the development of autonomous, adaptive mission planning control approaches is required. This document provides a literature research to support a thesis proposal which investigates a possible solution to this case. Specifically, the thesis proposal aims to contribute to the research field of SARS, by developing a mission planning approach that exploits the non-homogeneous nature of various UAVs in the system. Since the mission planning controller incorporates the individual capabilities of each agent into its mission strategy, it is hypothesised that such an approach increases the victim search efficiency of the SARS with respect to both the search time and area coverage. Furthermore, the mission planning controller is adaptive, since its strategies may improve based on input data that the agents gather during their deployment, such as unexpected fire hazards or damaged structures. Essentially, the controller progressively improves the fidelity of the environment model, which the controller uses to further optimise its search strategy. Based on this research, the literature study attempts to answer the following question:

> ”Can an adaptive, autonomous mission planning approach for non-homogeneous agents increase the victim search efficiency of a robotic search-and-rescue system?”

To answer this question, the proposed thesis research is divided into two main parts: designing the mission planning module, and developing an appropriate simulator environment to test it in. Correspondingly, in Chapter 2 we have provided a theoretical review of various control approaches, which are commonly used in SARS-related research. The chapter mainly focuses on fuzzy control, model-predictive control (MPC), and prioritised planning, with additionally a number of combined/hybrid control approaches. This is followed by Chapter 3, in which the current state-of-the-art of SARS research has been reviewed. With this survey, it has ben possible to identify a number of trends in the mission planning and agent hardware used in SARS research (or in more general applications). The first of these trends, is that airborne robots have taken over land-based robots as the dominant type of robotic SARS agent in the past decade. Furthermore, mathematical control approaches for mission planning are becoming popular in SARS research, although heuristic approaches are still considered viable for current applications. For almost all SARS agents, optical sensors such as cameras, heat sensors, and LiDAR form the primary sensor platform (and source of situational awareness). However, with an increase in wireless connectivity in today’s society, sensors that can detect wireless devices have been found viable for victim detection in a SAR setting. The final trend, is that uncertainty and disturbances in the SAR environment are rarely explicitly addressed in (theoretical) SARS research. Once the review on controller types and SARS research has been completed, it is necessary to consider the different aspects of search-and-rescue (SAR) simulation, which has been discussed in Chapter 4. More specifically, a simulator is considered as test bed, since the performance of any novel SARS is preferably tested in a safe, consequence-free environment before being applied to a real-life disaster scenario.

Following this comprehensive research of control approaches, the different aspects of SARS research, and SAR simulation, in Chapter 5 we have formulated the specific research proposal of this thesis work. First of all, it has been chosen to use a fuzzy-logic-assisted prioritised task assignment

and path planning (FLAPTAPP) approach for the mission planning controller. This control approach is chosen for its ability to incorporate expert (human) SAR knowledge in its control strategy, as well as its simple and computationally efficient structure. Additionally, the world model that the mission planning controller uses, is updated in real-time to provide the most accurate situational awareness. This enables the mission planning approach to adapt its strategies based on newly acquired information from the environment. The proposed control approach is next applied to a multi-agent, non-homogeneous SARS of UAVs, which are deployed in an indoor USAR environment. In an effort to validate the proposed SARS of this thesis work, a custom simulator is built based on previous work by the research group of this thesis work, and the internationally recognised SAR environment simulator by Robocup-Rescue. The motivation for this is that such an experiment yields a fairer comparison between different mission planning strategies, while still being feasible to construct in the time span of this thesis. The simulator includes SAR environment elements such as fire spread, victims, and structural damage. Both the modelling of fire spread and victim movement are of a stochastic nature, in order to simulate the uncertainty of their dynamics in a real-life SAR scenario. The different aspects of this thesis proposal are enveloped in the following research question:

> **Can an adaptive FLAPTAPP approach for non-homogeneous agents increase the victim search efficiency of a search-and-rescue system of UAVs in terms of victim search time and area coverage?**

The outcome of this thesis provides a more extensive mission planning approach, which exploits the individual capabilities of the agents in the SARS, thus contributing to non-homogeneous SARS research. Secondly, a novel control approach will be developed, termed FLAPTAPP, which combines both fuzzy task assignment and prioritised path planning in one controller. The main aim of combining these two elements, is that together they increase the victim search efficiency compared to other mission planning approaches, and thus contribute to the ultimate aim of SAR research of reducing disaster victims.

III

Appendix

A

Simulation Code Manual

All code for the numerical simulations developed during this thesis is made publicly available on the Github of the author[1]. This chapter explains how the code is constructed and how it should be used.

A.1. Prerequisites

This research is performed exclusively in MATLAB R2019b (academic use) on a PC with Intel Core i7 Processor with 2.20GHz frequency. In addition to this, two extra Matlab add-ons are required:

- **Global Optimization Toolbox**. This toolbox is required for the supervisory controller to perform the optimisation problem.
- **Parallel Computing Toolbox**. This toolbox reduces the time it takes to run the experiments by distributing the computational effort of both the local and supervisory controller over multiple parallel processors. This toolbox is not strictly necessary, however the following changes must be made to the code if it is not used:
  1. The `parfor` command in **`simulate_step.m`** - `line 79` must be changed to a regular `for` command.
  2. The optimisation options in **`MPC_optimise.m`** - `lines 57-59` must be commented out.

A.2. main.m

The file **`main.m`** is the main file that is run when conducting experiments/simulations.

`Lines 13-28` The file starts by clearing the work space and adding all necessary directory paths. Additionally, the user can toggle the Boolean variables `plot_scan` `plot_env`, which shows the global scan certainty map and a schematic representation of the environment during the simulation, respectively.

`Lines 31-63` Two user inputs are defined with preset dialog options, which determine the type of search approach used (`search_app`) and the type of simulation in which the SAR system is evaluated (`sim_type`). After this, `sim_type` is used to set the appropriate simulation settings

`Line 65` The loop for all selected simulation cases starts here. If either one of the special simulation cases is chosen, the code simply runs that case only (e.g. if the user has previously selected `'Case1'` to be run, the code will run this case and stop afterwards). If the `Random` simulation option is chosen, the code will run multiple seeded random simulation scenarios with seeds values from `start` until `stop`.

[1] https://github.com/dekoningchristopher/NonHomogenous-SARS-Mission-Planning

`Lines 66-86` The previously selected simulation is initialised with the function handle **`init_Environment.m`** (details in Appendix A.3). Furthermore, if selected previously, these lines create two figure windows for both the global scan certainty map and the environment.

`Lines 89-104` The loop for simulating the steps of a single simulation case starts here. The simulation is terminated if the maximum limit of `tau_max` time steps has been reached. In this loop, **`simulate_step.m`** is the primary function that simulates the next time step (detail in Appendix A.4). The updated scan certainty map and environment plots are subsequently shown.

`Lines 107-116` Once `tau_max` has been reached, the simulation is completed and the relevant raw data is stored in the folder `\_data` in a `.mat` file. The location and name of the data file is automatically generated from the simulation type and search approach. For the special simulation cases, the raw data includes:

- `tot_SCAN`, the summed scan certainty of all cells in the environment.
- `M_SCAN`, the global scan certainty map of the environment.
- `M_TRACK`, the tracks of each agent during the simulation.
- `M_VIC`, the global victim map.

In addition to these parameters, the random simulation cases also save `confl`, which is the total number of search conflicts registered during the simulation.

`Lines 119-126` In these lines the user can toggle which results they would like to plot. The primary plotting settings are contained in **`plot_results.m`**.

A.3. init_Environment

In **`main.m`** `line 44` the function handle `init_Environment` is created. This function handle is constructed from the user's input and calls the desired simulation function (`init_Random.m`, `init_Case1.m`, `init_Case2.m`, `init_Case3.m`, `init_Case4.m`, or `init_CombiCase.m`). In this section only `init_Random.m` will be extensively detailed, however all special simulation cases are roughly structured the same way.

`Lines 34-45` The first few lines construct the various environment maps with the desired environment dimensions. Additionally, these lines initiate the (empty) containers for the performance indicators. Finally, the function `init_fuzzy.m` constructs the Mamdani fuzzy inference system for the local controller (including the rule base and all input and output membership functions).

`Lines 57-67` The obstacles are randomly placed in the environment with seeded random numbers that are scaled to the environment dimensions. The number of obstacles is dependent on the obstacle density value `perc_obs`, which represents what percentage of the total environment is covered in obstacles.

`Lines 71-81` Similarly to the obstacles, a fixed number of victims is randomly placed in the environment with a (different) set of seeded random numbers. The victim location is stored in the occupancy map such that the agents can detect them. Additionally, the global victim map `M_VIC` is created, which has dimensions $(rows, cols) = (n_{vic}, 5)$. Columns 1 and 2 contain a victim's y and x coordinates, respectively. Column 3 contains the victim health state, which is a randomly initiated value in the range $[50, 100]$. Finally, column 4 indicates the first time of detection of a victim, and column 5 shows how many times a victim has been visited (this element remains '`NaN`' if a victim is not detected).

`Lines 85-87` Finally, the SAR agents are initiated and added to the environment. Each SAR agent can be accessed through the map container structure `A_database`, which is constructed with the function `get_agents.m`. This function fills the database with elements of the class `agent`. The `agent` class is used to store both fixed and variable properties/states of each simulated SAR agent. The fixed properties of a SAR agent include its name, sensory perception radius r_p, and uncertainty reduction rate η. The variable properties of an agent are its (x, y) coordinates, perception field $E_{\mathrm{a_i}}$, and a list of all victims it has found. With `get_agents.m` the SAR agents are initiated with a set of predefined properties, however they may be adjusted for the special simulation cases.

A.4. simulate_step.m

The function `simulate_step.m` contains all calculations that update the entire simulation for a single time step. This includes updating the victims, calculating the path/next action for each agent (based on the selected search approach), and finally updating their respective positions in the environment map.

`Line 44` Updates the victim position and health state with the function `update_victims.m`.

`Lines 47-52` Updates the perception field of all agents with `get_perception.m` and stores this new array in the agent database. Additionally, the global scan map and victim map are updated with this function, by increasing the scan certainty of the cells within $E_{\mathrm{a_i}}$ and checking if the agent has detected any victims. This simulates an agent scanning its surroundings.

`Line 55` This line initiates a switch case for the string variable corresponding to the selected search approach `search_app`. This determines which search approach is used to determine the next step for each SAR agent.

`Lines 56-70` For both the selfish and cooperative search approach, the local controller of each SAR agent is activated. First, each agent constructs its own local victim map `M_VIC_hat` using the list of victims it has already found (stored in `agent.vic_list`). This enables each agent to calculate the priority score for each cell within $E_{\mathrm{a_i}}$ with the function `calc_localprio.m`.

`Lines 73-81` For each cell in $E_{\mathrm{a_i}}$ the SAR agent calculates the K shortest paths with `plan_kshortest.m`. Within this function, the function `plan_path.m` is used to find the globally shortest path to the target cell using A* Search. The function `plan_path.m` is an adapted version of the A* Search tutorial provided by P. Premakumar[2].

`Lines 83-101` Next, the function `grade_path.m` grades each of the K paths. Due to the use of parallel processing, only a single one of these K paths can be stored to the overall list of paths and their corresponding grades (`P_i` and `GRADE_LIST`, respectively). Hence, an intermediate step is taking by storing all K paths and their grades in the variable `temp_gradelist`, and only adding the path with the highest grade to `P_i` and `GRADE_LIST`. Once all target-path combinations have been determined, the local solution for the SAR agent is obtained by determining the highest scoring path. The paths for all agents are contained in the cell `P`.

`Lines 106-118` Both the cooperative and selfish register search conflicts, by determining if any SAR agents have an intersecting perception field of more than `I` cells. The supervisory controller is activated only if the cooperative search approach has been selected. The supervisory controller performs the optimisation problem with the function `MPC_optimise.m`.

[2]https://www.mathworks.com/matlabcentral/fileexchange/26248-a-a-star-search-for-path-planning-tutorial

`Lines 120-123` In case of the pure-MPC search approach, the initial path solution must be randomly initialised with `init_path.m` before performing the optimisation procedure.

`Lines 125-130` In case of the ant-colony search approach, the path of each agent is determined with the function `ACS_search.m`.

`Lines 132-137` Similarly, `random_search.m` determines the paths of all agent in case the random search approach is selected.

`Lines 139-153` With the determined paths, the next step of each agent is determined by executing the first action/cell of their respective paths. The new position of the agents is updated in their class object and in the environment map. Furthermore, their old position must be removed from this map.

`Lines 156` Finally, this line determines the total scan certainty of the environment and stores it in the variable `tot_SCAN`.

B

Simulation Code Flowcharts

This chapter provides the flow charts for a number of sections within the code for the numerical simulations.

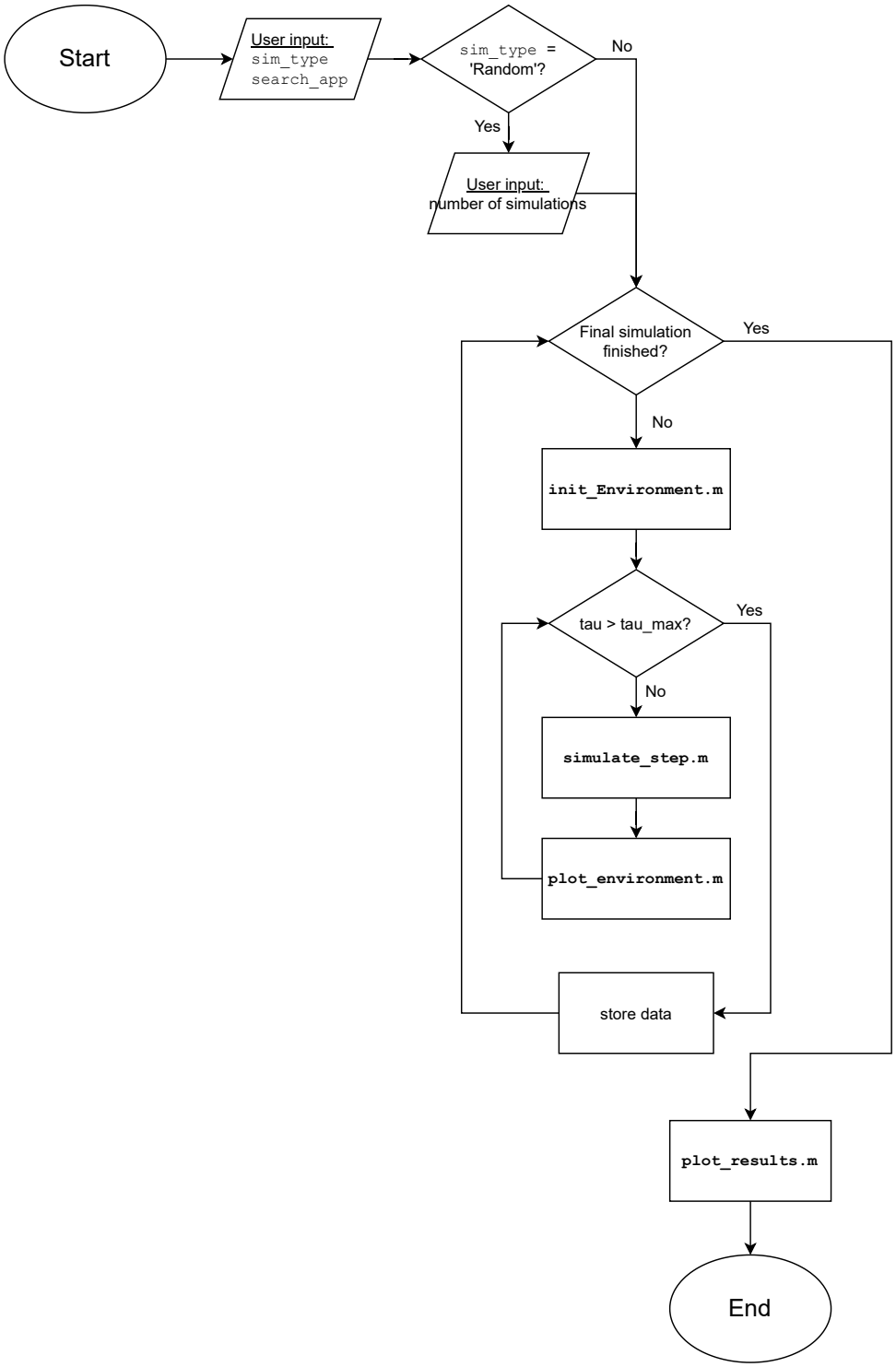

Figure B.1: Flowchart for file `main.m`

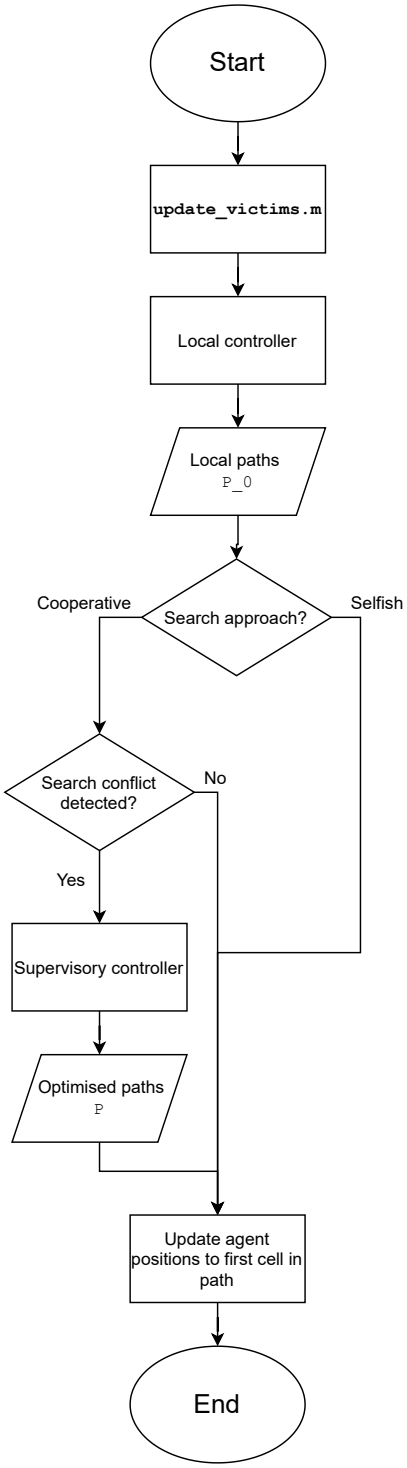

Figure B.2: Flowchart for file `simulate_step.m`

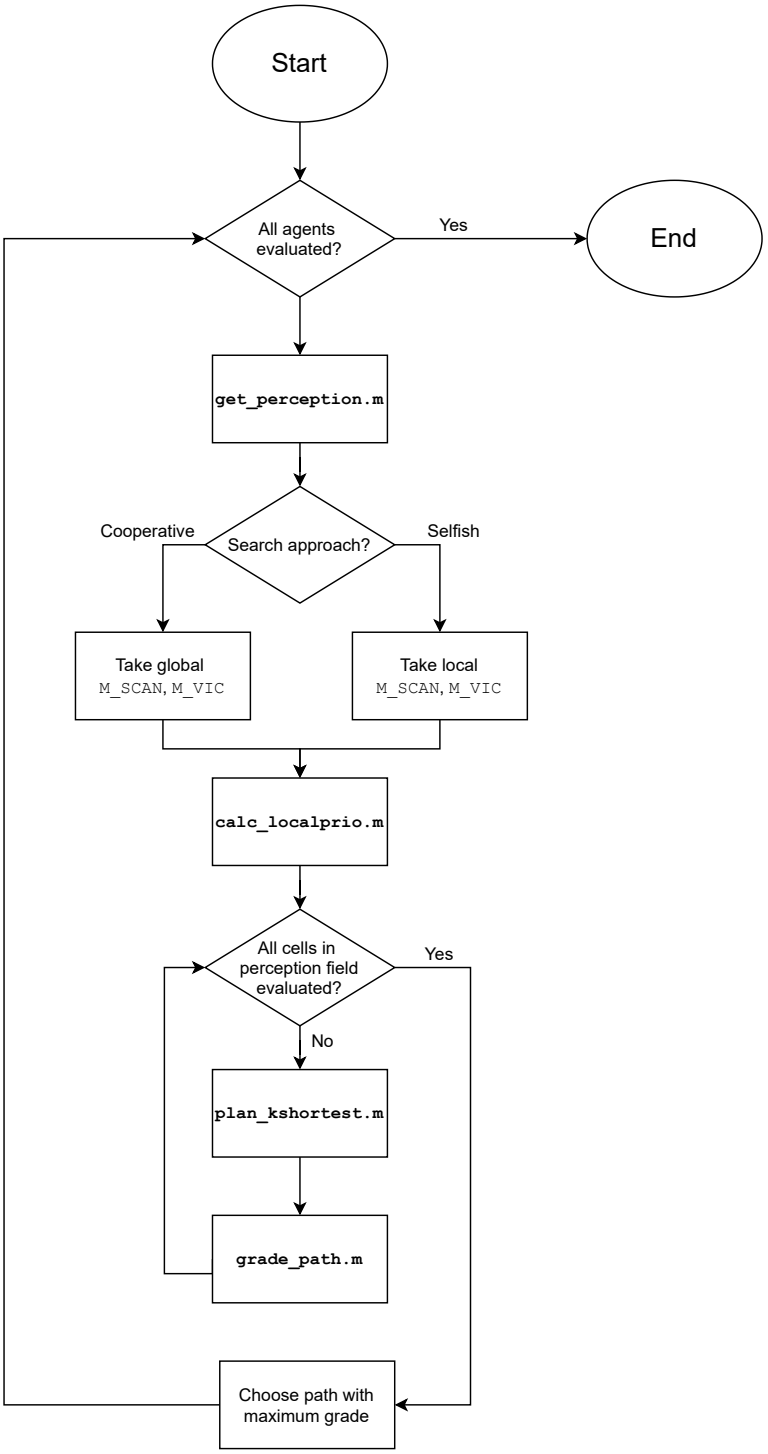

Figure B.3: Flowchart for agent's local controller code.

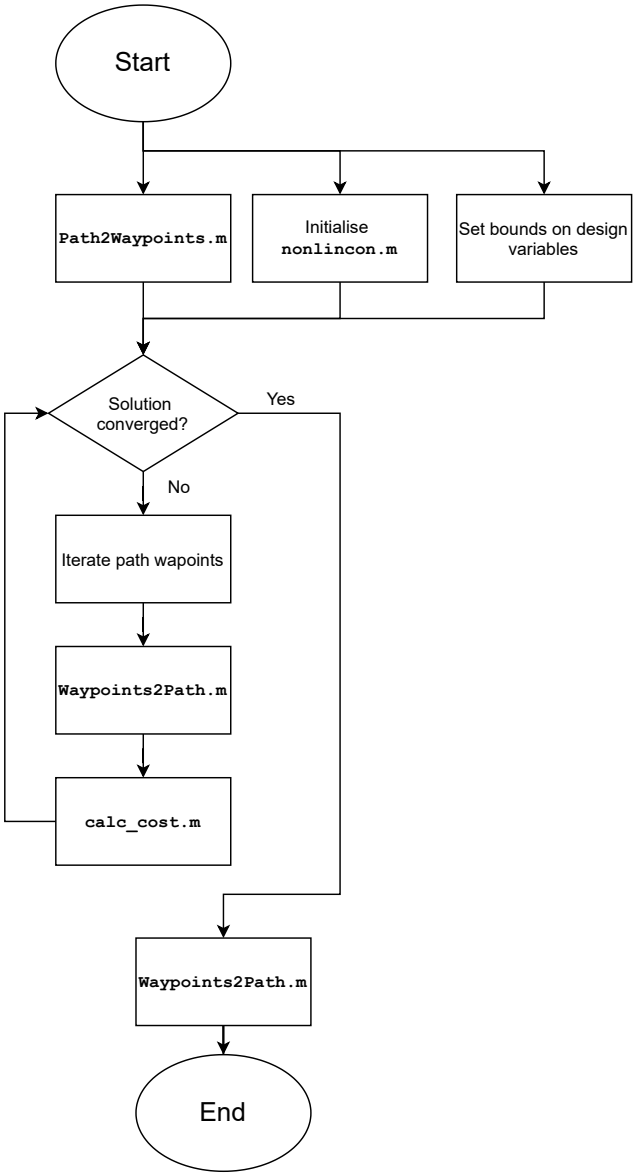

Figure B.4: Flowchart for control architecture's supervisory controller code.

Bibliography

[1] *Beirut explosion: Lebanon's government 'to resign' as death toll rises*. Aug. (2020). URL: https://www.bbc.com/news/world-middle-east-53720383 (visited on 01/22/2021).

[2] L. Alsaafin, R. Allahoum, and T. Regencia. *Endemic corruption caused Beirut blast, says Diab: Live updates*. Aug. (2020). URL: https://www.aljazeera.com/news/2020/08/10/endemic-corruption-caused-beirut-blast-says-diab-live-updates/ (visited on 01/22/2021).

[3] P. Wallemacq. *Economic losses, poverty & disasters: 1998-2017*. Centre for Research on the Epidemiology of Disasters, (2018).

[4] A. Khorram-Manesh. "Handbook of Disaster and Emergency Management", *Supported by DGE-CHO, Gothenburg, EU: Kompendiet* (2017), pp. 18–24.

[5] 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 (2003), pp. 367–385.

[6] A. W. Coburn, R. J. S. Spence, and A. Pomonis. "Factors determining human casualty levels in earthquakes: mortality prediction in building collapse", *Proceedings of the tenth world conference on earthquake engineering*. Vol. 10. Rotterdam, Netherlands. (1992), pp. 5989–5994.

[7] L. A. Zadeh. "Fuzzy sets", *Information and Control* Vol. 8 (1965), pp. 338–353.

[8] D. J. Dubois, H. Prade, and R. R. Yager. *Readings in fuzzy sets for intelligent systems*. Morgan Kaufmann, (2014).

[9] E. H. Mamdani. "Application of fuzzy algorithms for control of simple dynamic plant", *Proceedings of the institution of electrical engineers*. Vol. 121. 12. IET. (1974), pp. 1585–1588.

[10] T. Takagi and M. Sugeno. "Fuzzy identification of systems and its applications to modeling and control", *IEEE transactions on systems, man, and cybernetics* Vol. 1 (1985), pp. 116–132.

[11] A.J. Van der Wal. "Application of fuzzy logic control in industry", *Fuzzy Sets and Systems* Vol. 74, no. 1 (1995), pp. 33–41.

[12] Z. Gao, T. A. Trautzsch, and J. G. Dawson. "A stable self-tuning fuzzy logic control system for industrial temperature regulation", *IEEE Transactions on Industry Applications* Vol. 38, no. 2 (2002), pp. 414–424.

[13] H. Lee and S. Cho. "A new fuzzy-logic anti-swing control for industrial three-dimensional overhead cranes", *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*. Vol. 3. Seoul, South Korea. (2001), 2956–2961 vol.3.

[14] L. Karunarathne, J. T. Economou, and K. Knowles. "Fuzzy Logic control strategy for Fuel Cell/Battery aerospace propulsion system", (2008), pp. 1–5.

[15] H. Schulte and H. Hahn. "Fuzzy state feedback gain scheduling control of servo-pneumatic actuators", *Control Engineering Practice* Vol. 12, no. 5 (2004), pp. 639–650.

[16] M. Akalp, A. L. Dominguez, and R. Longchamp. "Supervisory fuzzy control of a rotary cement kiln", *Proceedings of MELECON '94. Mediterranean Electrotechnical Conference*. Vol. 2. Antalya, Turkey. (1994), pp. 754–757.

[17] M. Erdmann and T. Lozano-Perez. "On multiple moving objects", *Algorithmica* Vol. 2, no. 1-4 (1987), p. 477.

[18] M. Čáp, P. Novák, A. Kleiner, and M. Seleckỳ. "Prioritized planning algorithms for trajectory coordination of multiple mobile robots", *IEEE transactions on automation science and engineering* Vol. 12, no. 3 (2015), pp. 835–849.

[19] J. P. Van Den Berg and M. H. Overmars. “Prioritized motion planning for multiple robots”, *IEEE/RSJ International Conference on Intelligent Robots and Systems*. (2005), pp. 430–435.

[20] P. Velagapudi, K. Sycara, and P. Scerri. “Decentralized prioritized planning in large multirobot teams”, *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Edmonton, Alta. (2010), pp. 4603–4609.

[21] D. W. Clarke, C. Mohtadi, and P. S. Tuffs. “Generalized predictive control—Part I. The basic algorithm”, *Automatica* Vol. 23, no. 2 (1987), pp. 137–148.

[22] J. B. Rawlings, D. Q. Mayne, and M. Diehl. *Model predictive control: theory, computation, and design*. Vol. 2. Nob Hill Publishing Madison, WI, 2017.

[23] R. Scattolini. “Architectures for distributed and hierarchical model predictive control–a review”, *Journal of process control* Vol. 19, no. 5 (2009), pp. 723–731.

[24] S. Singer and J. Nelder. “Nelder-mead algorithm”, *Scholarpedia* Vol. 4, no. 7 (2009), p. 2928.

[25] R Babuska and J Kober. “Knowledge-Based Control Systems (lecture notes)”, (2019).

[26] S. J. Qin and T. A. Badgwell. “A survey of industrial model predictive control technology”, *Control engineering practice* Vol. 11, no. 7 (2003), pp. 733–764.

[27] Y. Wang and S. Boyd. “Fast model predictive control using online optimization”, *IEEE Transactions on control systems technology* Vol. 18, no. 2 (2009), pp. 267–278.

[28] N. Sandell, P. Varaiya, M. Athans, and M. Safonov. “Survey of decentralized control methods for large scale systems”, *IEEE Transactions on automatic Control* Vol. 23, no. 2 (1978), pp. 108–128.

[29] D. D. Šiljak. “Decentralized control and computations: status and prospects”, *Annual Reviews in Control* Vol. 20 (1996), pp. 131–141.

[30] A. Mesbah. “Stochastic model predictive control: An overview and perspectives for future research”, *IEEE Control Systems Magazine* Vol. 36, no. 6 (2016), pp. 30–44.

[31] J. R. Layne and K. M. Passino. “Fuzzy model reference learning control”, *Journal of Intelligent & Fuzzy Systems* Vol. 4, no. 1 (1996), pp. 33–47.

[32] W. R. Van Soest, Q. P. Chu, and J. A. Mulder. “Combined feedback linearization and constrained model predictive control for entry flight”, *Journal of guidance, control, and dynamics* Vol. 29, no. 2 (2006), pp. 427–434.

[33] Michael J Kurtz and Michael A Henson. “Input-output linearizing control of constrained nonlinear processes”, *Journal of Process Control* Vol. 7, no. 1 (1997), pp. 3–17.

[34] C. Sampedro, A. Rodriguez-Ramos, H. Bavle, A. Carrio, P. de la Puente, and P. Campoy. “A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques”, *Journal of Intelligent & Robotic Systems* Vol. 95, no. 2 (2019), pp. 601–627.

[35] S. Lee, H. Kim, and B. Lee. “An Efficient Rescue System with Online Multi-Agent SLAM Framework”, *Sensors* Vol. 20 (2020), p. 235.

[36] S. Grogan, R. Pellerin, and M. Gamache. “The use of unmanned aerial vehicles and drones in search and rescue operations - a survey”, *Proceedings of the PROLOG*. Hull, UK. (2018).

[37] R. Arnold, J. Jablonski, B. Abruzzo, and E. Mezzacappa. “Heterogeneous UAV Multi-Role Swarming Behaviors for Search and Rescue”, *IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA)*. Victoria, BC, Canada. (2020), pp. 122–128.

[38] Z. Beck, W. L. T. Teacy, N. R. Jennings, and A. C. Rogers. “Online planning for collaborative search and rescue by heterogeneous robot teams”, *Proceedings of the International Conference on Autonomous Agents & Multiagent Systems*. Singapore. (2016), pp. 1024–1033.

[39] H. Chenji, W. Zhang, M. Won, R. Stoleru, and C. Arnett. “A wireless system for reducing response time in Urban Search & Rescue”, *IEEE 31st International Performance Computing and Communications Conference*. Austin, TX. (2012), pp. 215–224.

[40] W. Stecz and K. Gromada. “UAV mission planning with SAR application”, *Sensors* Vol. 20, no. 4 (2020), p. 1080.

[41] V. San Juan, M. Santos, and J. M. Andújar. “Intelligent UAV map generation and discrete path planning for search and rescue operations”, *Complexity* Vol. 2018 (2018).

[42] M. A. Goodrich, L. Lin, B. S. Morse, and M. Roscheck. “Supporting wilderness search and rescue with integrated intelligence: autonomy and information at the right time and the right place”, *Association for the Advancement of Artificial Intelligence*. Atlanta, GA. (2010).

[43] M. A. Goodrich, B. S. Morse, D. Gerhardt, J. L. Cooper, M. Quigley, J. A. Adams, and C. Humphrey. “Supporting wilderness search and rescue using a camera-equipped mini UAV”, *Journal of Field Robotics* Vol. 25, no. 1-2 (2008), pp. 89–110.

[44] F. A. de Alcantara Andrade, A. Reinier Hovenburg, L. Netto de Lima, C. Dahlin Rodin, T. A. Johansen, R. Storvold, C. A. Moraes Correia, and D. Barreto Haddad. “Autonomous unmanned aerial vehicles in search and rescue missions using real-time cooperative model predictive control”, *Sensors* Vol. 19, no. 19 (2019), p. 4067.

[45] J. M. Riley and M. R 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. 3. Los Angeles, CA. (2004), pp. 693–697.

[46] X. Chen, H. Zhang, H. Lu, J. Xiao, Q. Qiu, and Y. Li. “Robust SLAM system based on monocular vision and LiDAR for robotic urban search and rescue”, *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. Shanghai, China. (2017), pp. 41–47.

[47] T.J. Tanzi, M. Chandra, J. Isnard, D. Camara, O. Sebastien, and F. Harivelo. “Towards ”Drone-Borne” Disaster Management: Future Application Scenarios.”, *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* Vol. 3, no. 8 (2016).

[48] Y. Liu and G. Nejat. “Robotic urban search and rescue: A survey from the control perspective”, *Journal of Intelligent & Robotic Systems* Vol. 72, no. 2 (2013), pp. 147–165.

[49] H. Durrant-Whyte and T. Bailey. “Simultaneous localization and mapping: part I”, *IEEE Robotics Automation Magazine* Vol. 13, no. 2 (2006), pp. 99–110.

[50] T. Bailey and H. Durrant-Whyte. “Simultaneous localization and mapping (SLAM): part II”, *IEEE Robotics Automation Magazine* Vol. 13, no. 3 (2006), pp. 108–117.

[51] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age”, *IEEE Transactions on Robotics* Vol. 32, no. 6 (2016), pp. 1309–1332.

[52] P. Schmuck and M. Chli. “Multi-UAV collaborative monocular SLAM”, *IEEE International Conference on Robotics and Automation (ICRA)*. Singapore. (2017), pp. 3863–3870.

[53] S. Ganesan, M. Shakya, A. F. Aqueel, and L. M. Nambiar. “Small disaster relief robots with swarm intelligence routing”, *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief*. Kollam, India. (2011), pp. 123–127.

[54] G. Loukas and S. Timotheou. “Connecting trapped civilians to a wireless ad hoc network of emergency response robots”, *11th IEEE International Conference on Communication Systems*. Singapore. (2008), pp. 599–603.

[55] W. Wang, R. Joshi, A. Kulkarni, W. K. Leong, and B. Leong. “Feasibility study of mobile phone WiFi detection in aerial search and rescue operations”, *Proceedings of the 4th Asia-Pacific workshop on systems*. Singapore. (2013), pp. 1–6.

[56] B. P. Gerkey and M. J. Matarić. “A formal analysis and taxonomy of task allocation in multi-robot systems”, *The International journal of robotics research* Vol. 23, no. 9 ((2004)), pp. 939–954.

[57] J. Berger and N. Lo. “An innovative multi-agent search-and-rescue path planning approach”, *Computers & Operations Research* Vol. 53 (2015), pp. 24–31.

[58] E. Rudnick-Cohen, J. W. Herrmann, and S. Azarm. “Risk-based path planning optimization methods for unmanned aerial vehicles over inhabited areas”, *Journal of Computing and Information Science in Engineering* Vol. 16, no. 2 (2016).

[59] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjou, and S. Shimada. “RoboCup Rescue: search and rescue in large-scale disasters as a domain for autonomous agents research”, *IEEE International Conference on Systems, Man, and Cybernetics*. Vol. 6. Tokyo, Japan. (1999), pp. 739–743.