TU Delft

Delft University of Technology

Hierarchical Integration of Model Predictive and Fuzzy Logic Control for Combined Coverage and Target-Oriented Search-and-Rescue via Robots with Imperfect Sensors

de Koning, Christopher; Jamshidnejad, Anahita

DOI
10.1007/s10846-023-01833-2

Publication date
2023

Document Version
Final published version

Published in
Journal of Intelligent and Robotic Systems: Theory and Applications

Citation (APA)
de Koning, C., & Jamshidnejad, A. (2023). Hierarchical Integration of Model Predictive and Fuzzy Logic Control for Combined Coverage and Target-Oriented Search-and-Rescue via Robots with Imperfect Sensors. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *107*(3), Article 40. https://doi.org/10.1007/s10846-023-01833-2

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright
Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy
Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

This work is downloaded from Delft University of Technology.
For technical reasons the number of authors shown on this cover page is limited to a maximum of 10.

Journal of Intelligent & Robotic Systems (2023) 107:40
https://doi.org/10.1007/s10846-023-01833-2

SHORT PAPER

Hierarchical Integration of Model Predictive and Fuzzy Logic Control for Combined Coverage and Target-Oriented Search-and-Rescue via Robots with Imperfect Sensors

Christopher de Koning[1] · **Anahita Jamshidnejad**[1]

Received: 11 January 2022 / Accepted: 7 February 2023 / Published online: 14 March 2023
© The Author(s) 2023

Abstract

Search-and-rescue (SaR) in unknown environments is a crucial task with life-threatening risks. SaR requires precise, optimal, and fast decisions to be made. Robots are promising candidates expected to execute various SaR tasks autonomously. While humans use heuristics to effectively deal with uncertainties of SaR, optimisation of multiple objectives (e.g., the mission time, the area covered, the number of victims detected), in the presence of physical and control constraints, is a mathematical challenge that requires machine computations. Thus including both human-inspired and mathematical capabilities in decision making of SaR robots is highly desired. However, developing control approaches that exhibit both capabilities has been significantly ignored in literature. Moreover, coordinating the decisions of the robots in large-scale SaR missions with affordable computation costs is an open challenge. Finally, in real-life, due to defects (e.g., in the sensors of the robots) or environmental factors (e.g., smoke) data perceived by SaR robots may be prone to uncertainties. We introduce a hierarchical multi-agent control architecture that simultaneously provides the following advantages: exploiting non-homogeneous and imperfect perception capabilities of SaR robots; improving the global performance as it is provided by centralised controllers; computational efficiency and robustness to failure of the central controller as offered by decentralised control methods. The integrated structure of the proposed control framework allows to combine human-inspired and mathematical decision making methods, via respectively fuzzy logic and model predictive control, in a coordinated and computationally efficient way. Our results for various computer-based simulations show that while the area coverage with the proposed control approach is comparable to existing heuristic methods that are particularly developed for coverage-oriented SaR, our approach has a significantly better performance regarding locating the trapped victims. Furthermore, with comparable computation times, the proposed control approach successfully avoids conflicts that may appear in non-cooperative control methods. In summary, the proposed multi-agent control system is capable of combining coverage-oriented and target-oriented SaR in a balanced and coordinated way.

Keywords Multi-robot search-and-rescue · Model predictive control · Fuzzy logic control · Imperfect sensors

1 Introduction

Search-and-rescue (SaR) robots are expected to take over life-threatening tasks, especially within initial stages of searching an unknown environment, in order to reduce the risks for the SaR crew. This will allow human resources to be available for other tasks, e.g., logistics and assisting the detected victims [1, 2]. Moreover, SaR robots should reduce the crucial time of finding the trapped victims. Robots can move through areas that are inaccessible to humans, gather information (e.g., about the location of victims, explosive materials, debris) and make maps of the environment. This way SaR robots contribute to improving the situational awareness of SaR crews, which is essential for mitigating the mission risks and for saving the lives of the trapped victims [3–6].

SaR is categorised into target-oriented and coverage-oriented, based on its objectives. In target-oriented SaR, the distribution of the targets within the environment is initially known (see, e.g., [7–11]). When the SaR environment

✉ Anahita Jamshidnejad
A.Jamshidnejad@tudelft.nl

[1] Control and Operations Department, Delft University of Technology, Delft, The Netherlands

is unknown, coverage-oriented approaches are used [12]. Ant colony algorithms are bio-inspired area coverage methods that are computationally efficient and easy to implement [13, 14]. Machine learning and neural network methods are also used for area coverage, where robots progressively learn effective area coverage behaviours [15, 16]. The main drawback of such methods is their need for being trained before they can be implemented. To address this issue, autonomous learning-based methods, including generalised model-free reinforcement learning, have been developed (see, e.g., [17, 18]). Via such algorithms, the system keeps on learning online an optimal policy. Although promising, autonomous learning methods face computational challenges in real-life SaR implementations. This is due to the large size and varying dynamics of SaR environments, which make the learning procedure more complicated. Moreover, in SaR missions there may be high risks associated with implementing a solely learning-based algorithm, especially before the system achieves an optimal (or close to optimal) performance. More specifically, during the stages that the algorithm is learning an optimal policy, there are serious risks regarding losing the trapped victims or delaying their detection, which may result in severe health issues or even fatalities.

Most coverage-oriented approaches do not systematically incorporate victim or target detection in their search behaviour. Arnold et al. in [19] present a cooperative, multi-agent SaR system with the objective of both victim detection and exploration. The SaR agents, however, are steered according to fixed behaviour sets. This limits the adaptability and thus efficiency of these robots in highly dynamic SaR environments. The majority of the existing SaR control methods are either coverage-oriented or target-oriented. Moreover, model predictive control (MPC), which is an optimisation-based control method that systematically incorporates state and input constraints, and that provides robustness to uncertainties has been ignored for the crucial task of area coverage in SaR [9]. Instead, MPC has mainly been used for reference tracking in target-oriented SaR in (partially) known environments (see, e.g., [7, 20, 21]).

In order to speed up mapping the SaR area and to reduce the risk of mission failure, a fleet of SaR robots may be deployed (see, e.g., [9, 22, 23]). In centralised multi-agent SaR control, robots are controlled via a centralised system that determines the mission plans for all these robots (see, e.g., [8–11]). In decentralised multi-agent SaR control, local (or on-board) controllers are used for the robots (see [13, 15, 19, 24, 25]. Best et al. [26] present a cooperative distributed information gathering approach for SaR robots where based on learning and heuristics robots visit stationary, pre-known goal regions. While a task assignment problem is solved in a communication-wise efficient way, there are no (dynamic) uncertainties involved in the environment of the robots. Otte et al. [27] address a cooperative task-assignment problem using a decentralised auction approach. In particular, the effect of lossy communication among the agents on the performance of the multi-agent system is investigated, with the aim of providing insight into the selection of an auction algorithm that, despite lossy communication, satisfies the desired performance criteria of a multi-agent system. A multi-agent search-planning approach is introduced in [28] for wilderness SaR with a team of aerial and ground robots. In their approach, the initial trajectory planning for the aerial robots is performed offline. After an aerial robot detects a (possibly moving) target, the robot tracks it until a ground robot intercepts this target.

While decentralised control approaches are more robust to failure and are computationally more efficient than centralised approaches, providing reliable and stable communication among the robots and missing a global vision of the entire system are challenges of decentralised control approaches [24]. Hierarchical architectures can combine the strengths of centralised and decentralised control methods (see, e.g., [29]). Particularly, for multi-robot SaR systems hierarchical control architectures can provide coordination in the behaviour of local controllers. However, a limited amount of research on hierarchical control for SaR robots is available. Examples include [30–32], which are all limited to target-oriented SaR.

Currently, SaR robots need (intensive) supervision and control from human operators. On the one hand, for safety, efficiency, and avoiding additional challenges regarding online human-robot interaction there is interest in making these robots autonomous [1, 3, 25, 33, 34]. On the other hand, humans use their heuristics effectively in order to deal with uncertainties of SaR missions. Therefore, providing SaR robots with human knowledge will improve their performance. Human knowledge, which is provided as information-based control for SaR robots in [35], was shown to improve the performance of SaR robots in finding the targets. Thus having both capabilities of human-inspired decision making and mathematical control is highly desired for SaR robots. However, control approaches that exhibit both capabilities have been ignored significantly in literature.

In this paper, we introduce a hierarchical control architecture for multi-agent control of SaR robots with non-homogeneous, imperfect sensors that combines mathematical and human-inspired control methods in a computationally efficient way.

The main contributions of this paper include:

1. Introducing a novel hierarchical control framework for multi-objective control and coordination of multi-robot systems and for exploiting their non-homogeneous sensor imperfections in unknown environments: The

resulting control system benefits both from robustness to failure and computational efficiency of decentralised control methods and from globally effective performance of centralised control methods.
2. Integrating human-inspired and mathematical decision making by formulating local fuzzy logic controllers, which mimic decision making of human experts, and a supervisory model predictive control (MPC) system, which provides mathematical precision in the decisions of the system, systematically handles state and input constraints, improves the global performance of the multi-robot system based on its optimal and predictive decision making, and resolves conflicts of local heuristic controllers.
3. Implementing the proposed control framework for combined coverage and target-oriented SaR via multi-robot systems with imperfect sensors for optimising the mission time, area coverage, and number of detected victims, and running extensive experiments via computer-based SaR simulations in order to evaluate various performance criteria (e.g., computational efficiency, percentage of the area covered, overall certainty level of the map developed for the SaR environment, number of victims detected) of the proposed SaR control methods compared to the state-of-the-art methods.

Additionally, since the local controllers steer the robots, the multi-robot control system will not fail due to a failure of the centralised controller. In that case, as our simulation results indicate, the remaining decentralised control system can still steer the SaR system safely, although with a degraded performance. Finally, our novel control approach and formulation for multi-agent SaR control systems enable MPC to provide all its strong points (including (sub)optimality, systematic incorporation of both state and input constraints, and robustness to SaR uncertainties) for, not only target-oriented, but also coverage-oriented SaR in unknown areas.

The rest of the paper is structured as it follows. In Section 2 the problem formulation is detailed. Section 3 discusses the proposed hierarchical mission planning control approach for SaR robots. Section 4 describes the case study and experimental setup and presents, analyses, and discusses the results. Section 5 concludes the paper and provides suggestions for future research.

2 Problem Formulation

In this section, we explain and formulate the details of the mission planning problem of SaR robots with non-homogeneous imperfect sensors. In particular, we discuss the modelling of the SaR environment and victims and the uncertainties involved, as well as the mathematical formulation of the perception capabilities of SaR robots.

2.1 SaR Environment

The SaR environment E is modelled by a bounded, discretised, 2-dimensional cellular area of $L_x \times L_y$ cells (see Fig. 1). Each cell in the SaR environment corresponds to the coordinates (x, y) of its centre and may be vacant or occupied by a static obstacle (i.e., wall, pillar, rubble), or by a victim and/or a SaR robot. A cell can embed a single victim at a time. Moreover, obstacles make a cell inaccessible for SaR robots and for victims. The following uncertainties exist for SaR robots:

- External uncertainty regarding the SaR environment, i.e., the total number of victims and obstacles and their positions are unknown
- External, random uncertainty about the pattern of movement of the victims
- Internal (i.e., structural) and external (i.e., proximal) uncertainties regarding the perceived data

An $L_x \times L_y$ matrix $\mathcal{W}(\kappa)$, called the *occupancy map*, is used to record cells that are occupied by static obstacles after being detected by a SaR robot. Furthermore, whenever a victim is detected by the sensor of SaR robot i, the robot stores the location, perceived health state, and time of detection of the victim in a local matrix (specific to the robot) called the *victim map* $\mathcal{V}_i(\kappa)$ of SaR robot a_i. This map is used by the controller of the robot to make the current control decision. However, for the sake of efficiency for the on-board computations and the memory storage, the robots - via their local victim maps - only keep track of those victims who have been selected as a target by the controller and have been visited by the robot (i.e., the robot has been in the same cell as the victim), as well as of those victims who are currently within the perception field of the robot. Thus SaR robots do not record any memory of those victims who have previously been detected by the robot, but have not been selected as a target for the robot.

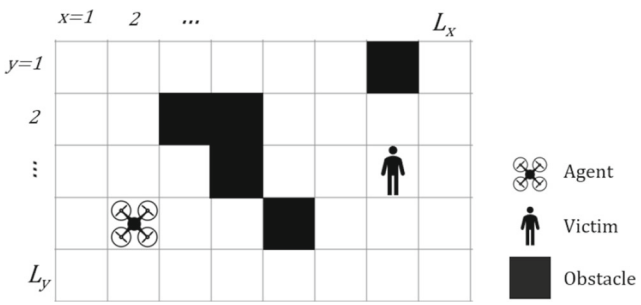

Fig. 1 Schematic view of a SaR environment

The scan certainty $c(x, y, \kappa)$ of cell (x, y) for time step κ is a value within [0, 1] that specifies the certainty level regarding the information available about cell (x, y). Moreover, each cell (x, y) corresponds to a proximal uncertainty at time step κ that is a function of the Euclidean distance of the cell from all SaR robots that scan the cell. The scan certainty of a cell depends on whether or not the cell has been scanned by any SaR robots and if so, how accurate the perceived data is, i.e., the scan certainty depends on the proximal uncertainty. This relationship is explained in detail in Section 2.2. Initially the scan certainty of all cells within the SaR environment is zero. The scan certainty for all cells is included in an $L_x \times L_y$ matrix $\mathcal{C}(\kappa)$, called the *scan certainty map*, which will be updated in time.

2.2 SaR Robots

We consider a multi-robot SaR system composed of N agents a_i $(i = 1, \ldots, N)$ that, per simulation time step, may move to one of the 8 neighbouring cells (see Fig. 2(a)). These robots are equipped with optical cameras and sensors that localise the victims and that assess their health state (e.g., acoustic and heat sensors [1, 36] or sensors that detect WiFi-enabled devices [37]). The perception field $E_i(\kappa)$ of SaR robot i for time step κ includes all cells of the SaR environment that fall within a circle of radius $r_{\mathrm{p},i}$, centred at the position of the robot at time step κ, where $E_i(\kappa) \subseteq E$ (see Fig. 2(b)).

The data perceived by SaR robots may in general be imperfect, i.e., scanning a cell does not necessarily yield full certainty about the information within the cell. Two sources of uncertainty regarding the perceived data are considered: (1) *Structural imperfection*, which corresponds to a fixed perceptual uncertainty reduction rate $\eta_i \in (0, 1]$ per SaR robot a_i. More specifically, every time SaR robot a_i scans a cell, the uncertainty regarding the information of the cell is reduced by rate η_i. Thus when $\eta_i = 1$, there is no structural imperfection. Moreover, we do not consider sensors that are completely out of function (i.e., $\eta_i = 0$) due to structural imperfection. (2) *Proximal uncertainty*, which implies that while all cells within the perception field E_i of SaR robot a_i are scanned, the degree of increase in the scan certainty of these cells decreases according to their distance from the sensor.

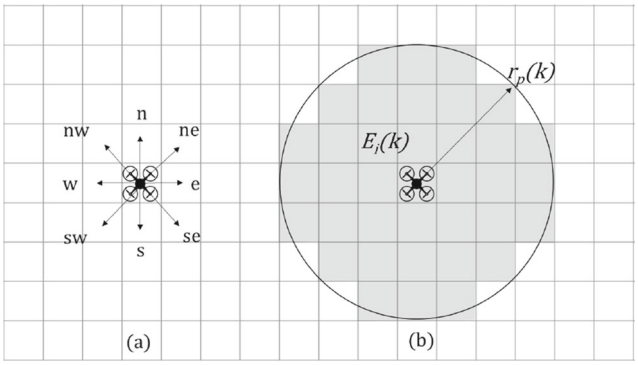

Fig. 2 (a) Movement possibilities of a SaR robot. (b) Perception field of a SaR robot

The structural imperfection of sensors and the proximal uncertainty of the cells together will result in an uncertainty dynamic ratio $\bar{\sigma}(x, y, \kappa)$ corresponding to every cell (x, y) per time step κ. We have:

$$z(x, y, \kappa + 1) = \bar{\sigma}(x, y, \kappa) z(x, y, \kappa) \tag{1}$$

with $z(x, y, \kappa)$ the scan uncertainty (i.e., $1 - c(x, y, \kappa)$) assigned to cell (x, y) at time step κ. Moreover, we have:

$$\bar{\sigma}(x, y, \kappa) = \prod_{i=1}^{N} \sigma_i(x, y, \kappa) \tag{2}$$

$$\sigma_i(x, y, \kappa) = 1 - (1 - \eta_i) e^{-r_i(x,y,\kappa)} \cdot \frac{1 - \mathrm{sign}\left(r_i(x, y, \kappa) - r_{\mathrm{p},i}\right)}{2} \tag{3}$$

where $r_i(x, y, \kappa)$ is the Euclidean distance of SaR robot a_i to cell (x, y) at time step κ, sign$(\cdot)$ represents the sign function, and $\sigma_i(x, y, \kappa)$ is the share of the uncertainty dynamic ratio of cell (x, y) at time step κ that is provided by the sensor of SaR robot a_i. The updated scan certainty for cell (x, y) is given by:

$$c(x, y, \kappa + 1) = 1 - z(x, y, \kappa + 1) \tag{4}$$

Based on Eq. 3, the effect of the proximity on the uncertainty dynamic ratio of the cells is modelled by an exponential function. More specifically, when $r_i(x, y, \kappa) = 0$, i.e., for the cell where SaR robot a_i is currently located at, the uncertainty dynamic ratio corresponding to SaR robot a_i is η_i (i.e., the maximum possible improvement in the scan certainty of the cell that can be provided by the sensor of SaR robot a_i). This uncertainty dynamic ratio varies exponentially until for $r_i(x, y, \kappa) \geq r_{\mathrm{p},i}$, it becomes unity (i.e., the scan certainty of the cell at the current time step does not improve as a result of a contribution of the sensor of SaR robot a_i).

SaR robots may differ from each other in two properties regarding their sensors: (1) The sensors of SaR robots may have different perception radii $r_{\mathrm{p},i}$ for $i = 1, \ldots, N$. (2) The accuracy of these sensors, and thus their perceptual uncertainty reduction rate η_i may be different.

2.3 Victim Modelling

The number, location, and health state of the victims are initially unknown for the SaR robots. The victims follow a random pattern of movement, i.e., victim v with position $(x_v^{\mathrm{v}}(\kappa), y_v^{\mathrm{v}}(\kappa))$ at time step κ may remain

in its current cell with probability p^{s} or may move to one of its (unblocked) neighbouring cells with a total probability $1 - p^{\mathrm{s}}$, which results in an equal probability of $p_v^{\mathrm{m}}(\kappa) = (1 - p^{\mathrm{s}})/n_v^{\mathrm{free}}(x_v^{\mathrm{v}}(\kappa), y_v^{\mathrm{v}}(\kappa), \kappa)$ to move to each of the (unblocked) neighbouring cells, where $n_v^{\mathrm{free}}(x_v^{\mathrm{v}}(\kappa), y_v^{\mathrm{v}}(\kappa), \kappa)$ corresponding to victim v is the number of free neighbouring cells for cell $(x_v^{\mathrm{v}}(\kappa), y_v^{\mathrm{v}}(\kappa))$ at time step κ. Note that for the sake of simplicity we consider the probability p^{s} to be constant in time and space, and for all victims. In case a more detailed model is desired, this probability may vary in time, and per cell and victim.

Moreover, each victim holds a certain health state, $h_v(\kappa)$, which varies within $[0, 100]$ and implies how healthy or injured the victim is at time step κ. Whenever a victim is detected by a SaR robot, their initial health state is registered. Over time, the health state of each victim may decrease with the rate $\Delta h_v(\kappa)$ given by:

$$\Delta h_v(\kappa) = \begin{cases} -\alpha & h^{\mathrm{crit}} \leq h_v(\kappa) \leq 100 \\ \beta h_v(\kappa) - \gamma & 0 \leq h_v(\kappa) \leq h^{\mathrm{crit}} \end{cases} \tag{5}$$

with $\alpha, \beta, \gamma > 0$, h^{crit} the critical health state, and $\gamma \geq \beta h^{\mathrm{crit}}$. Based on Eq. 5, a victim has a uniformly deteriorating health state whenever their health state is not less than h^{crit} (i.e., health state is stable), while the rate of deterioration of the health state becomes linear as soon as the health state is below h^{crit}. The updated health state is given by:

$$h_v(\kappa + 1) = \max\{h_v(\kappa) + \Delta h_v(\kappa), 0\} \tag{6}$$

A SaR robot detects a victim whenever they are both in the same cell. Without considering the technical details regarding data analysis, sensor fusion, or soft sensing in this paper, we assume that the robot detects the victim (see, e.g., [38, 39]) and assesses the health state of the victim, e.g., using a combination of WiFi, optical, thermal, or acoustic sensors and using image processing algorithms or via direct feedback received from the victims when possible (see, e.g., [40]).

3 Hierarchical Control System

Next we explain the proposed hierarchical control system that steers the search behaviour of SaR robots. The control architecture includes two levels (see Fig. 3): The lower level of control is composed of decentralised controllers that steer the local search behaviour of each SaR robot, while the higher control level includes a centralised supervisory controller that coordinates the behaviour of the decentralised controllers, such that search conflicts among SaR robots are resolved. SaR robots only communicate with the supervisory control level, without sharing any information among themselves. The proposed control architecture thus combines the strengths of centralised and decentralised control approaches.

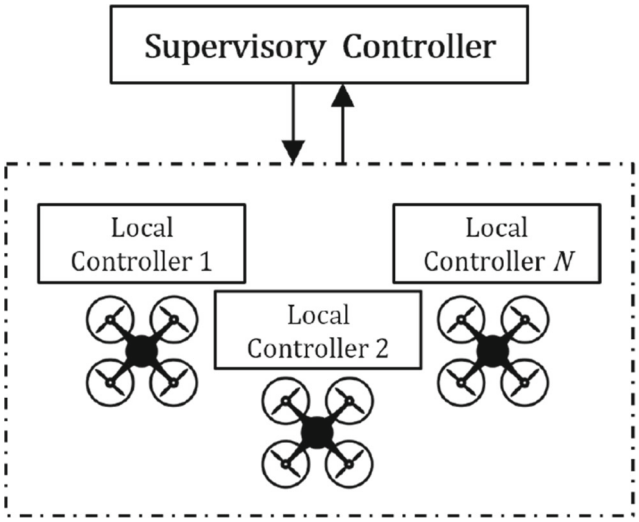

Fig. 3 Architecture of the proposed hierarchical, cooperative mission planning controller

3.1 Local Fuzzy Logic Controllers

At the local level, a SaR robot first processes the data that is captured via its sensors and then constructs a local priority map for its perception field. This map includes quantities corresponding to the importance of visiting the cells for the SaR robot. Next the local controller of the robot determines a path that yields the highest local gain according to a quantity called the path grade. Since the main objective of the SaR mission is to optimise the area coverage and the time efficiency of detecting the victims, the following two main criteria are considered in grading a path:

1. **Time reduction:** Each SaR robot should reach its targets in the least possible time, in order to contribute to minimising the overall mission time.
2. **Exploration increase:** Each SaR robot should scan as many (unexplored) cells as possible along its path, in order to contribute to maximising the overall area coverage.

These two criteria may possess a conflict, since for the first criterion the search behaviour should be target-oriented (in other words the robot should find the shortest possible path that leads it to the target as quickly as possible), while for the second criterion the search behaviour is coverage-oriented (in other words the robot should visit more cells before reaching its target). Therefore, the local controllers are developed such that a balanced trade-off between these criteria is provided.

3.1.1 Search Priority Assignment

The local controllers of SaR robots should first assign priorities to their potential paths, specifying the urgency for scanning every cell within the environment. For every cell in the perception field E_i of SaR robot a_i, a priority score is determined using a rule-based fuzzy logic control (FLC) method. The main motivation for using FLC is its computational efficiency, which is essential for local controllers due to the limited computational power available on board for SaR robots, and the capability of FLC in mimicking human's logic in decision making, which allows to incorporate human expert knowledge within the local controllers. Thus local controllers effectively mimic the reasoning of human experts without their direct supervision.

Fuzzy rules with the following formulation are used by local controllers of SaR robots:

$$\begin{aligned}\mathcal{R}_m: \quad & \textbf{If } e^{\mathrm{v}}(x,y,\kappa) \text{ is } A_{m,1} \textbf{ and } h_v(\kappa) \text{ is } A_{m,2} \\ & \textbf{and } c(x,y,\kappa) \text{ is } A_{m,3} \textbf{ then } \rho(x,y,\kappa) \text{ is } B_m\end{aligned} \tag{7}$$

with $e^{\mathrm{v}}(x,y,\kappa)$ the probability of existing a victim in cell $(x,y) \in E_i$ at time step κ, $m = 1,\ldots,M$ with M the number of rules, and $A_{m,1}$, $A_{m,2}$, $A_{m,3}$, and B_m fuzzy sets that adopt a linguistic term.

The fuzzy inference system corresponding to the rules given by Eq. 7 receives 3 inputs per cell (x,y) (i.e., the probability of existence of a victim in the cell, the health state of the potential victim, and the most recent scan certainty value of the cell) and assigns a search priority $\rho(x,y,\kappa)$ to cell (x,y) for time step κ. Note that every SaR robot a_i has access to its local knowledge stored in the local scan certainty map $\mathcal{C}_i(\kappa)$ and local victim map $\mathcal{V}_i(\kappa)$. The probability $e^{\mathrm{v}}(x,y,\kappa)$ of existence of a victim in cell (x,y) estimated by SaR robot a_i depends on the robot's sensor, and adopts either a very small value when the sensor receives no signal that implies a victim exists in the cell (for a sensor with $\eta_i = 1$ this value may be 0, while for a sensor with $\eta_i \in (0,1)$ a small positive value may be considered), or a percentage determined according to the structural imperfection and proximal uncertainty explained in Section 2.2. Based on Eq. 7, those cells within the perception field of the robot, where it is more likely to find a victim with a worse health state and that have not been (extensively) scanned yet will receive a higher priority. Inaccessible cells within the occupancy map $\mathcal{W}(\kappa)$ receive a null priority.

3.1.2 Path Planning

After prioritising the cells, each local controller determines potential paths for the corresponding SaR robot. In order to optimise the time, shortest paths are favourable, while for optimising the area coverage, paths that visit more cells with higher priorities are preferred. Thus the local controller applies an A* search approach [41] based on Yen's algorithm [42] to determine a certain number of shortest paths that end at every cell within the perception field of the robot. Afterwards these paths are graded based on their *travel time* and *degree of exploration* to specify how favourable they are for the SaR mission at the current time step. The travel time is computed based on the path length and the robot's speed. We suppose that a SaR robot moves one cell per time step, thus the travel time corresponds to the path length only. The degree of exploration of every potential path $P_i(\kappa) \subseteq E_i$ for SaR robot a_i at time step κ is computed via:

$$\epsilon\left(P_i(\kappa)\right) = \sum_{k=\kappa}^{\kappa+\ell(P_i(\kappa))-1} \lambda^k \rho\left(x_i^{\mathrm{a}}(k), y_i^{\mathrm{a}}(k), k\right) \tag{8}$$

where the path is defined by:

$$\begin{aligned}P_i(\kappa) = \Big\{ & \left(x_i^{\mathrm{a}}(\kappa), y_i^{\mathrm{a}}(\kappa)\right), \ldots, \\ & \left(x_i^{\mathrm{a}}\left(\kappa+\ell(P_i(\kappa))-1\right), y_i^{\mathrm{a}}\left(\kappa+\ell(P_i(\kappa))-1\right)\right)\Big\}\end{aligned} \tag{9}$$

with $\ell(P_i(\kappa))$ the path length, $\lambda \in [0,1]$ the discount factor, and $\rho(x_i^{\mathrm{a}}(k), y_i^{\mathrm{a}}(k), k)$ the priority value of cell $(x_i^{\mathrm{a}}(k), y_i^{\mathrm{a}}(k))$, which the robot should visit at time step $k = \kappa,\ldots,\kappa+\ell(P_i(\kappa))-1$ when it follows path $P_i(\kappa)$. Note that since the priority values for the cells corresponding to time steps $k > \kappa$ are based on the predictions/estimates, a discount factor is considered in order to reduce the potential influence of errors in these predictions/estimates. Finally, the grade of path $P_i(\kappa)$ is computed by (with $c_1, c_2 > 0$ constant values):

$$g\left(P_i(\kappa)\right) = -c_1 \ell\left(P_i(\kappa)\right) + c_2 \epsilon\left(P_i(\kappa)\right) \tag{10}$$

Remark 1 Since the paths that will be generated by the local controllers of the SaR robots are rectilinear, for practical implementations and to make it easier for real robots to execute these paths, we propose smoothening the paths before implementation (see, e.g., [43] for equations that can be used to smoothen such paths).

3.2 Supervisory MPC Controller

At the supervisory level, a centralised MPC-based controller is used that receives the local information corresponding to each SaR robot and that merges this information to build up global maps of the current perception fields of the robots. Note that while robots erase the non-target victims from their local victim maps (see Section 2.1 for details), the global victim map keeps track of all locally perceived information. This is practically possible because the global maps are recorded on a remote computer station that is not restricted by computational and memory limits.

The supervisory controller is called whenever a search conflict is identified, i.e., whenever the cardinality of the intersection of the perception fields of two SaR robots exceeds a certain threshold: card $\left(E_i \cap E_j\right) > \tau_{\text{int}}$. A model of the environment including the most updated cognitive maps is used as the prediction model of the supervisory controller, which determines globally optimal (within the controller's prediction time window) paths for the SaR robots. This optimality is defined as a trade-off among various objectives including the mission time, the area coverage, and the chances of visiting more trapped victims with a more crucial health state. Despite providing globally optimal solutions, the MPC controller is computationally demanding due to the size of the centralised optimisation problem and the non-linearities involved in the problem. Therefore, we provide the supervisory controller with the paths that are determined by the local controllers as a warm start for the MPC optimisation problem, in order to help the optimiser to converge faster to an optimal solution. Taking into account the objectives of the SaR mission, the objective function to be maximised by the supervisory controller at time step κ is given by:

$$J\left(\mathbb{P}(\kappa)\right) = w_1 \sum_{i=1}^{N} g\left(P_i(\kappa)\right) + w_2 \sum_{(x,y)\in E} c\left(x, y, N^{\text{p}}(\kappa)\right) \tag{11}$$

with $\mathbb{P}(\kappa)$ (the optimisation variable) the set of paths for all the N SaR robots and w_1 and w_2 constant weights. The objective function given by Eq. 11 is a weighted sum of two terms: (i) the overall grade of all paths (estimated by Eq. 10) and (ii) the total predicted scan certainty of the SaR environment at the end of the current prediction horizon $N^{\text{p}}(\kappa)$, which is given by $N^{\text{p}}(\kappa) = \max_{i=1,\ldots,N} \ell(P_i(\kappa))$. Thus the second term steers the fleet of the SaR robots to spread out over the environment. In other words, the supervisory controller provides a balanced trade-off between locally preferred paths per robot and globally optimal paths from the point of the area coverage. The supervisory controller does this using a global scan certainty map $\mathcal{C}(\kappa)$ of the environment and a global victim map $\mathcal{V}(\kappa)$, which are built by merging the local maps of all SaR robots.

The supervisory control optimisation problem for time step κ is given by (where the prediction window is $\{\kappa, \ldots, N^{\text{p}}(\kappa) - 1\}$):

$$\max_{\mathbb{P}(\kappa)} J\left(\mathbb{P}(\kappa)\right)$$

such that:

$$\mathbb{P}(\kappa) = \{P_1(\kappa), \ldots P_N(\kappa)\} \tag{12a}$$

For all the paths, Eq. (9) holds, with (12b)

$$\left(x_i^{\text{a}}(\kappa), y_i^{\text{a}}(\kappa)\right) \in E \setminus \mathcal{W}(\kappa)$$
$$\left(x_v^{\text{v}}(\kappa), y_v^{\text{v}}(\kappa)\right) \notin P_i(k) \cap P_j(k), \text{ where} \tag{12c}$$
$$i, j = 1, \ldots N, i \neq j, v = 1, \ldots, N^{\text{v}}(\kappa)$$
$$\left(x_i^{\text{a}}(k), y_i^{\text{a}}(k)\right) = \left(x_i^*(\kappa), y_i^*(\kappa)\right) \qquad i = 1, \ldots N \tag{12d}$$

Constraints (12a) and (12b) define the optimisation variable, and state that the paths should be feasible. Constraint (12c) restricts multiple SaR robots to visit the same victim, where $N^{\text{v}}(\kappa)$ is the number of victims detected until simulation time step κ. This constraint improves the victim search efficiency and the area coverage. To reduce the conservativeness of the problem and to avoid infeasibility, constraint (12c) may be defined as a chance (instead of a hard) constraint. Finally, constraint (12d) allows the starting point of the paths to be the most recent measured coordinates $(x_i^*(\kappa), y_i^*(\kappa))$ of the corresponding SaR robot.

Remark 2 Since the objective function of the supervisory MPC-based controller is defined in Eq. 11 as a weighted sum of the multiple control objectives, these objective terms will be normalised when implementing the optimisation problem.

4 Case Study

Next we discuss the results of computer-based simulations that are systematically designed to evaluate the performance of the proposed hierarchical control approach in comparison with state-of-the-art approaches for SaR. The simulations are implemented via MATLAB R2019b on a PC with Intel Core i7 Processor with 2.20 GHz frequency. Whenever an optimisation problem should be solved to determine the paths of the SaR robots, the path planning problem is solved using pattern search as optimisation method, since this algorithm showed to be faster than other alternative approaches. In order to make sure that the resulting paths meet the requirements of a discrete cellular environment for the numerical simulations (i.e., the way points defining the path have to be located at the centre of a cell) the continuous coordinates for the way points determined by pattern search are projected to the centre of the cells using the round function. For the parameters of the algorithms, we did a manual tuning with respect to the default settings.

4.1 Simulation Setup

We consider the following four search approaches that are common for SaR, and compare their performance, in terms of victim detection, area coverage, and computational efficiency, with the proposed hierarchical control approach, which we call **cooperative controller** due to the supervisory MPC level.

Selfish Controller A control system composed of the local controllers described in Section 3.1, where the main difference with the cooperative controller is the lack of a supervisory controller. These controllers make decisions that fit their own circumstances only.

Pure MPC Controller An optimisation-based search approach with the MPC structure of the supervisory controller described in Section 3.2, where the main difference with the cooperative controller is the lack of warm starting with trajectories that are proposed by the local controllers. Instead, as it is common in the implementation of MPC, the pure MPC controller receives the shifted solution of the previous time step as a warm start (see [44] for more details). Note that, in order to account for the non-convexity of the problem, we ran the simulations for the pure MPC controller with multiple starting points within the given time budget. However, the results for the pure MPC controller with warm start were the best. Thus in this paper only those results for the pure MPC controller that correspond to the warm start have been presented.

ACS Controller A heuristic ant-colony-based search approach based on [13], where the global scan certainty map $\mathcal{C}(\kappa)$ is used for pheromone map for the ant colony system.

Exhaustive Controller A random search strategy for SaR robots. Note that, in practice, such random search strategies are commonly used as a reference base for other search methods.

Remark 3 Ideally a centralised MPC controller can provide the desired performance for a system by providing a globally optimal solution. This requires to provide enough computational resources and time for the centralised MPC controller. However, a main challenge that needs to be addressed for SaR problems is to provide a balanced trade-off between performance and computation time, such that the control system meets the real-time requirements of a SaR robotic team. Therefore, we are interested in assessing how well different control approaches can steer the behaviour and performance of the SaR system when they are constrained by the computation time. Thus for both the pure MPC controller and the supervisory MPC controller we have considered a limited time budget, which may in some cases imply a degradation of the performance to meet the given computation time.

A set of 20 simulation scenarios, each lasting 300 simulation time steps, with a seeded random placement of victims and obstacles in an environment of a fixed

Table 1 Modelling and control parameters

Parameter	Value
L_x	40
L_y	25
Number of victims	25
p^{s}	0.6
α	0.25
β	1/60
γ	1
h^{crit}	30
λ	0.6
c_1	2.0
c_2	5.0
τ_{int}	30
w_1	1.0
w_2	0.05

Table 2 Rule base of the fuzzy inference system (with the inputs given by e^{v}, i.e., the probability of existence of a victim, h_v, i.e., the health state of the potential victim, and c, i.e., the scan certainty of the cell to be visited, and the output is ρ, i.e., the search priority of the cell)

$\mathcal{R}_m$	e^{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

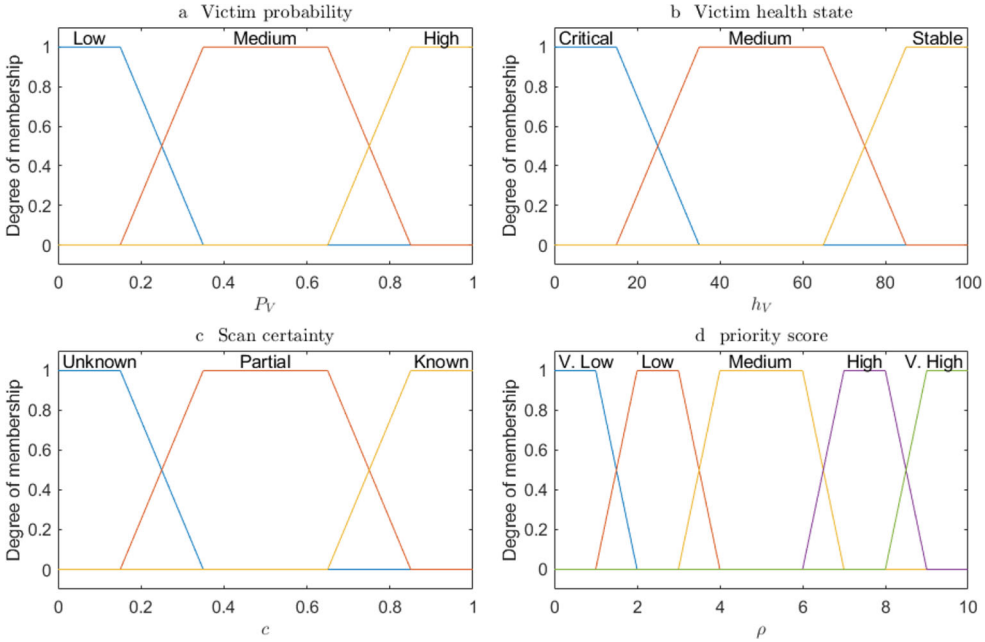

Fig. 4 Trapezoidal fuzzy membership functions defined for the inputs and outputs of the Mamdani rule bases

size is considered. The parameters required for these simulations to estimate the movement of the victims and to compute (5) and (8) are given in Table 1. The coefficients/weights in Eqs. 10 and 11 are also given in Table 1, where the corresponding values are tuned manually via extensive experiments. The terms that describe the sets $A_{m,1}$, $A_{m,2}$, $A_{m,3}$, and B_m in Eq. 7 should for real-life scenarios be deduced from real expert knowledge. For the numerical simulations designed in our case studies, we have defined the corresponding rule base based on intuition. More specifically, the sets $A_{m,1}$, $A_{m,2}$, and $A_{m,3}$ are verbally described by, respectively, "Low, Medium, High", "Critical, Medium, Stable", and "Unknown, Partially Known, Known". This selection allows us to build up a Mamdani rule base composed of 27 rules. For the output set B_m we select one of the following terms, "Very Low, Low, Medium, High, Very High", based on intuition and suited for the given realisations of the input fuzzy sets. The resulting Mamdani rule base is represented in Table 2. The corresponding membership functions used in Eq. 7 are shown in Fig. 4, where trapezoidal functions have been selected, since they have proven to result in good quality control systems in various real-life applications [45].

For the case study, we consider 2 SaR robots with different sensory perception radii and perceptual uncertainty reduction rates. The robots start at fixed coordinates without any initial information about the SaR environment. Thus all maps are initialised to zero/null. The parameters used for the SaR robots are shown in Table 3. These parameters have been selected such that the influence of non-homogeneity and imperfection of the sensors can properly be incorporated into the numerical simulations.

In order to evaluate various search approaches in terms of the area coverage, victim search efficiency, and computational efficiency, the following performance metrics have been considered. The area coverage is assessed

Table 3 Parameters of the SaR robots

	$r_{\mathrm{p},i}$	η_i	$(x^{\mathrm{a}}_{i,0}, y^{\mathrm{a}}_{i,0})$		$r_{\mathrm{p},i}$	η_i	$(x^{\mathrm{a}}_{i,0}, y^{\mathrm{a}}_{i,0})$
General case				**Case 3**			
$i=1$	6	0.1	(1,16)	$i=1$	7	0.1	(9,10)
$i=2$	4	0.3	(13,25)	$i=2$	3	0.3	(5,8)
Case 1				**Case 4**			
$i=1$	6	0.1	(10,8)	$i=1$	4	0.1	(8,7)
$i=2$	4	0.3	(10,6)	$i=2$	4	0.3	(8,9)
Case 2				**Case 5**			
$i=1$	6	0.1	(6,10)	$i=1$	7	0.1	(9,17)
$i=2$	4	0.3	(6,8)	$i=2$	3	0.3	(5,15)

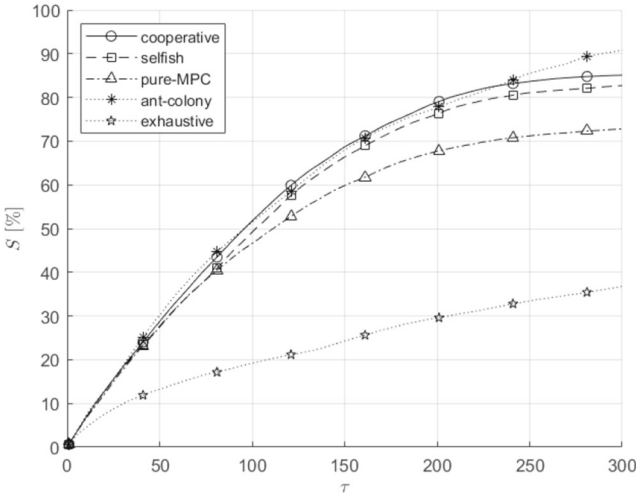

Fig. 5 Total scan certainty corresponding to different controllers as a percentage of the maximum scan certainty that can be obtained for the SaR environment

via two performance metrics: (1) Total scan certainty of the environment as a function of the simulation time step, i.e.:

$$S(\kappa) = \sum_{(x,y)\in E} c(x, y, \kappa) \tag{13}$$

(2) Rise time for the total scan certainty (of a particular percentage). These performance metrics quantify the absolute area coverage, as well as the speed, thus efficiency, of each search approach. The victim search efficiency is evaluated via three metrics: (1) Number of (live and deceased) victims detected per simulation time step. (2) Simulation time step for which each victim is detected. (3) Health state of each victim at the time of detection. Finally, the average time for making control decisions per simulation time step is used to report the computational effort of each control approach.

4.2 Results & Discussions

Figure 5, Table 4, Fig. 6, and Table 5 represent the results of the simulations including, respectively, the total scan certainty, the number of simulation time steps required per control approach in order to reach a total scan certainty of 50%, 70%, 80%, 85%, and 90%, the victim detection efficiency, and the average decision making time. Moreover, Fig. 7 shows the number of the simulation time steps when search conflicts have been registered for the cooperative and selfish controllers. For the cooperative controller, the number of the conflicts registered is the number of the times that the supervisory MPC controller has been activated.

Table 4 Number of the simulation time steps needed to reach certain degrees of scan certainty

Degree of scan certainty	50%	70%	80%	85%	90%
Cooperative controller	97	157	208	291	–
Selfish controller	104	167	238	–	–
ACS controller	97	159	218	249	292
Pure MPC controller	112	229	–	–	–
Exhaustive controller	–	–	–	–	–

Figure 5 shows that the selfish and cooperative controllers achieve a comparable total scan certainty of above 80% by the end of the simulations (to be more accurate, with a slightly higher percentage (around 85%) for the cooperative controller). However, Table 4 shows that the cooperative controller is significantly faster in reaching particular levels of scan certainty in later stages of the simulation (e.g., the cooperative controller needs 14.4% less time to reach an 80% total scan certainty). The ACS controller (see Fig. 5) reaches an overall scan certainty that is 6.71% larger than that of the cooperative controller, with a comparable rise time in earlier stages of the simulation. This is because the objective function of the ACS controller solely considers the scan certainty map per time step to determine the most favourable next step for each SaR robot. Thus the ACS controller has a single objective (thus no objective conflicts) as opposed to the selfish and cooperative controllers. Based on Fig. 5 and Table 4, the pure MPC controller performs much better than the exhaustive controller, but worse than the cooperative, selfish, and ACS controllers. This is mainly because the pure MPC controller is prone to falling within a sequence of local optima since it relies on the solution of the previous time steps as warm start. This not only highlights the importance of using more systematic (i.e., in line with the global objectives of the SaR mission) warm starts for the MPC controller, but also stresses that the exploratory nature of the selfish and cooperative controllers plays an important role in avoiding such issues. Finally, the exhaustive controller shows the worst performance regarding the area coverage, due to its lack of systematic search objective.

Figure 6(a) and (b) show that the cooperative and selfish controllers achieve a similar result regarding the number of the victims found (i.e., 25), and using these controllers correspond to the least number of victims deceased. However, considering all the simulation runs, the cooperative controller has lower variances, i.e., 0.134 and 0.345, for respectively the number of victims found and deceased than those (i.e., 3.10 and 2.22) of the selfish controller. This indicates a more consistently satisfactory performance for victim search using the cooperative controller. Both the selfish and cooperative controllers outperform the pure MPC, ACS, and exhaustive controllers in terms of the victim detection, with the ACS and the exhaustive controller showing the worst performance (see Fig. 6(a) and (b)). This is because systematic victim detection is not an objective for these controllers. More

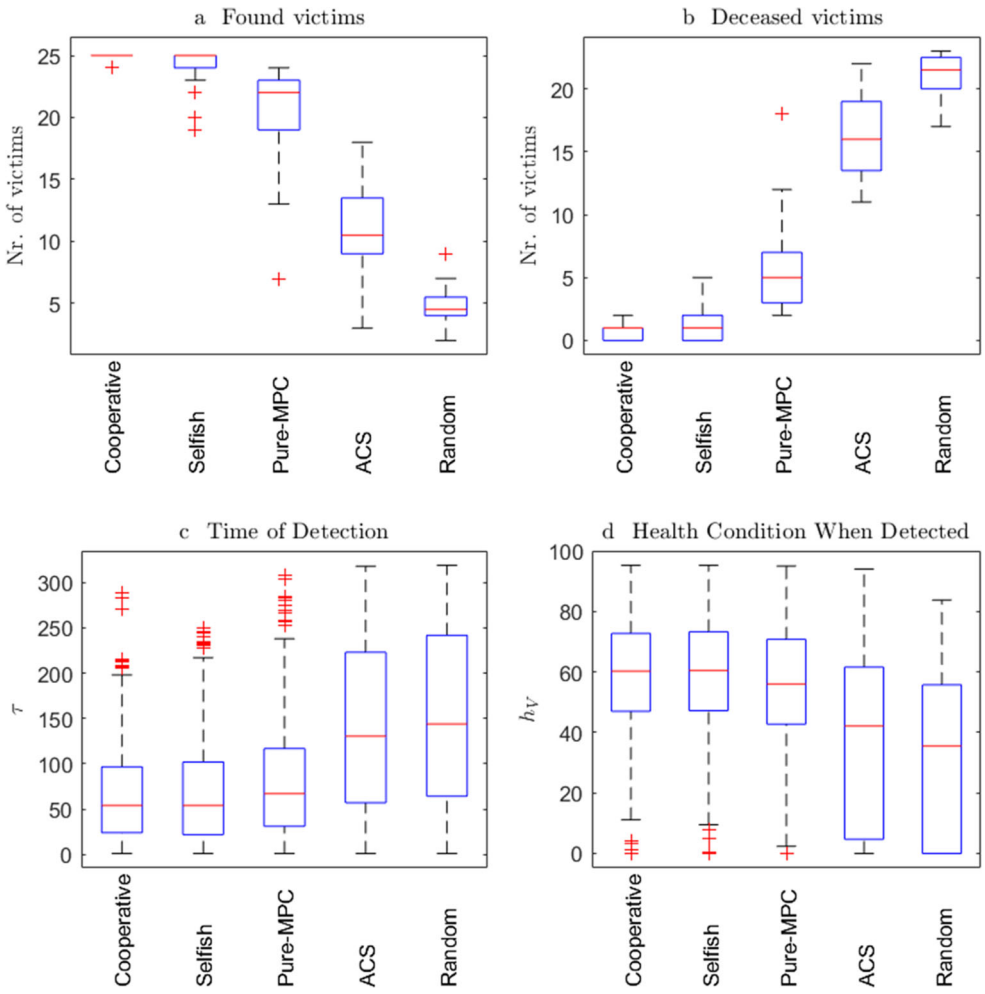

Fig. 6 Victim detection efficiency for each control approach evaluated by (a) number of the victims found, (b) number of the victims deceased, (c) time of detection, and (d) health state of the victims when detected

specifically, with the ACS and exhaustive controllers SaR robots may detect the victims randomly. The fact that the ACS controller detects more victims than the exhaustive controller is an indirect influence of its higher area coverage (see Fig. 5). While the pure MPC controller outperforms both the ACS and the exhaustive controller, compared to the cooperative and the selfish controller less victims are detected and more victims are deceased. This is due to the lower area coverage by the pure MPC controller, which has a negative impact on the victim detection efficiency. Based on Fig. 6(c) and (d) the cooperative and selfish controllers perform equally well considering the detection time and the health state of the victims. While the MPC controller detects less victims than the cooperative and selfish controllers, the detection time and the health state of the victims found are at comparable levels (more precisely, they are slightly worse) as those of the cooperative and selfish controllers.

Table 5 Average computation time for decision making per simulation time step for different controllers

Cooperative controller	Selfish controller	ACS controller	Pure MPC controller	Exhaustive controller
4.5 [s]	3.2 [s]	0 [s]	8.6 [s]	0 [s]

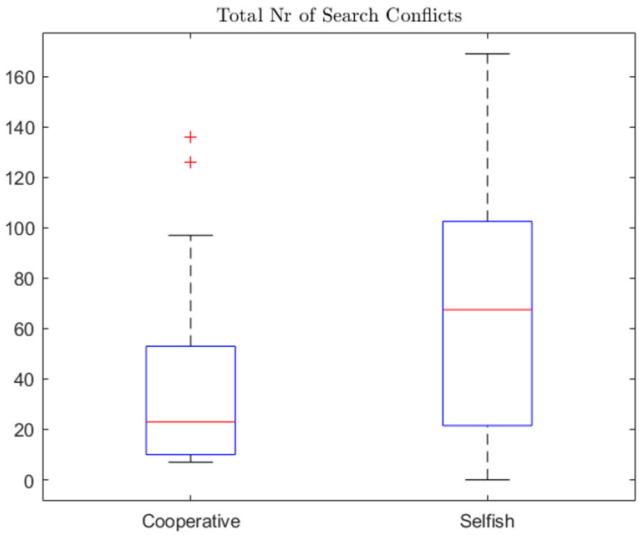

Fig. 7 Number of simulation time steps when conflict is detected for cooperative and selfish controllers

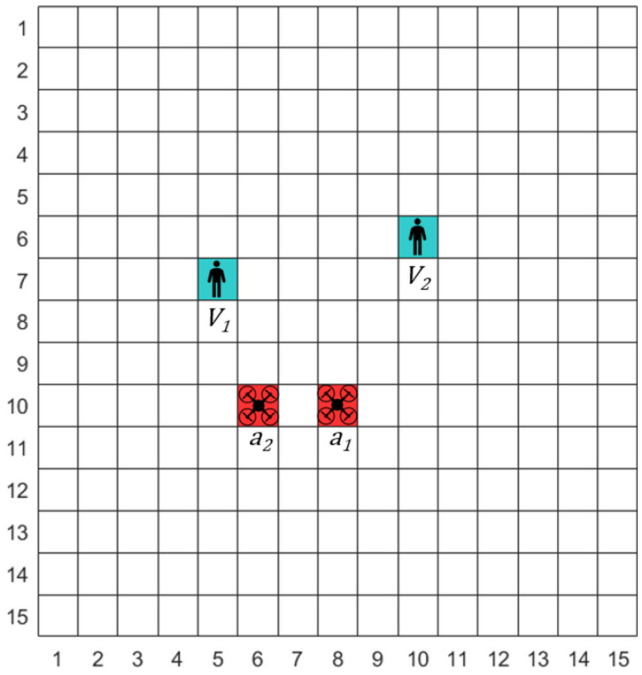

Fig. 8 Case 1: Conflict in the victim detection

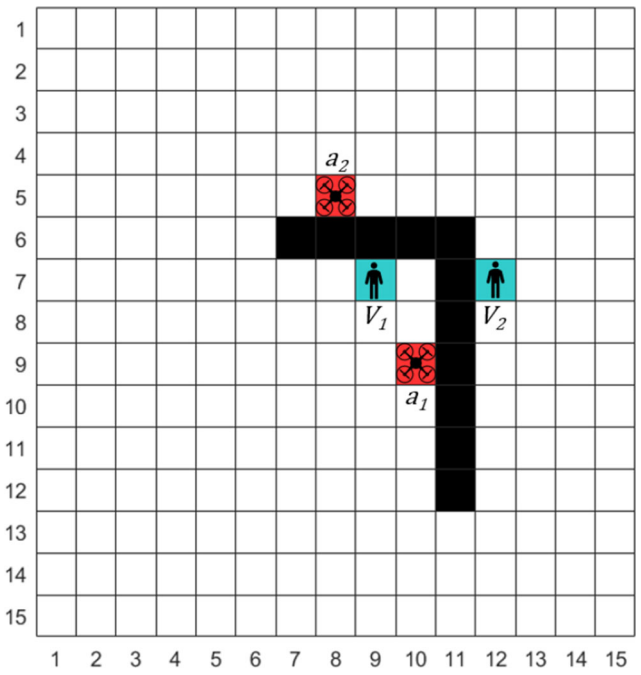

Fig. 10 Case 3: Exploitation of the perception fields

Finally, the computation time per decision (see Table 5) for the cooperative controller (i.e., 4.5 s) is almost half of the computation time when only MPC is used (i.e., pure MPC controller) to steer the system. Moreover, compared to the average decision making time of the selfish controller (i.e., 3.5 s), and considering the significantly better performance of the cooperative controller, this controller is the best choice among all the given controllers. Based on Fig. 7, compared to the selfish controller, search conflicts happen less when the cooperative controller is used. Thus each time the supervisory controller is triggered, by improving the global performance of the SaR system, it reduces the number of the future search conflicts.

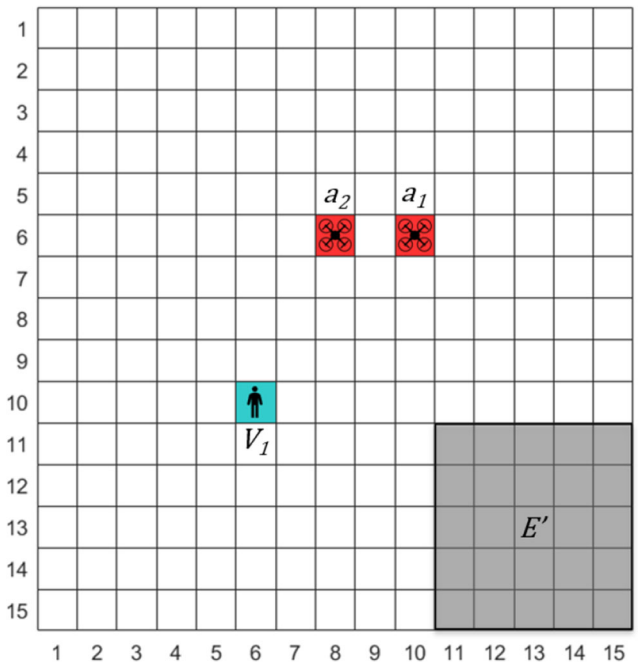

Fig. 9 Case 2: Conflict in the victim detection and area coverage

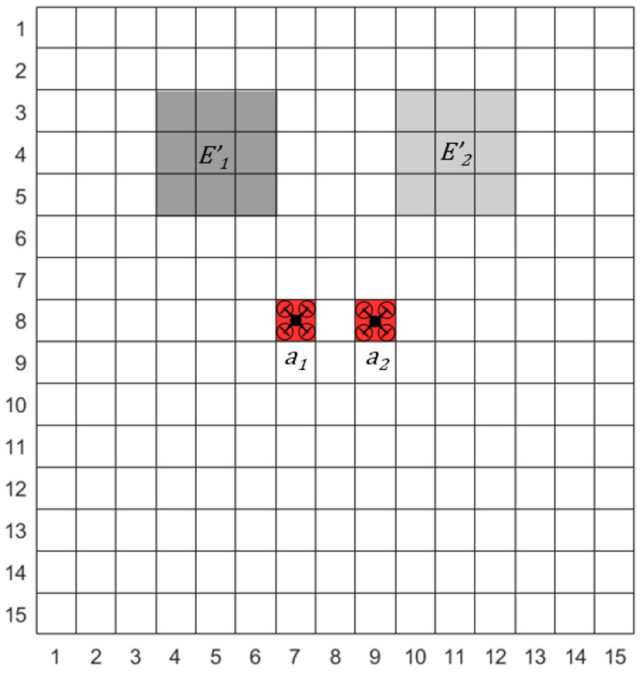

Fig. 11 Case 4: Exploitation of the sensor accuracies

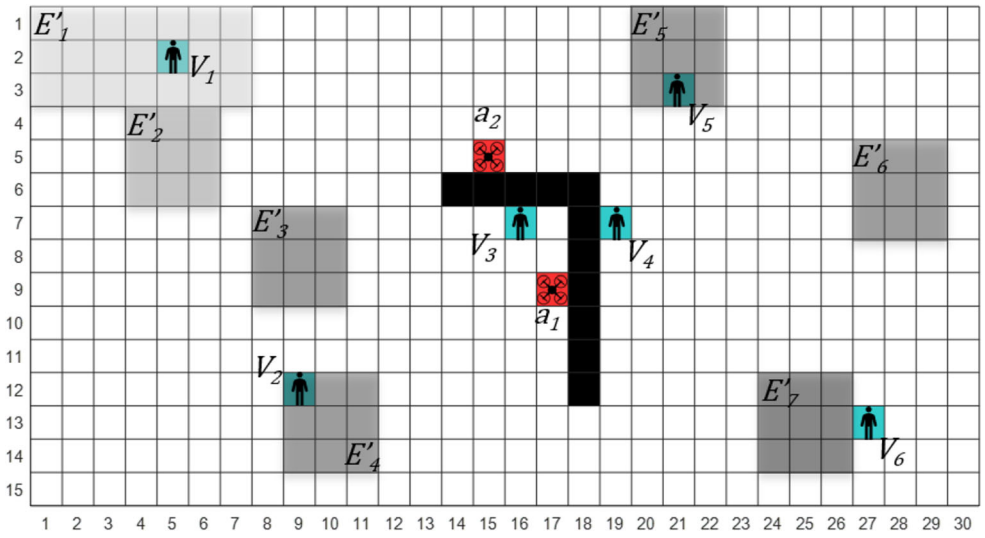

Fig. 12 Case 5: Combined scenario

Based on the results and discussions given above, the cooperative controller significantly outperforms the other methods. The next best controller is the selfish controller, that is the decentralised control system that remains when the supervisory MPC controller is excluded. These results further confirm the robustness of the proposed control architecture to failure of the supervisory MPC controller, i.e., while the performance degrades after the supervisory MPC controller is excluded from the control architecture, the performance of the SaR robotic team is still better than the other control methods used in the case study.

4.3 Structured Simulation Scenarios

In order to further assess the performance of the cooperative controller in a more structured way and to assess the problem solving behaviour of the controller when several types of conflicts among the local controllers exist, five cases of conflicts in smaller scales than the previous simulation scenarios are considered (see Figs. 8, 9, 10, 11 and 12).

Case 1. Conflict in the Victim Detection Consider 2 SaR robots and 2 victims in a partially known environment of size 15×15 with $c(x, y, 0) = 0.2$ for all $(x, y) \in E$ (see Fig. 8). The health states of the victims are 10 and 50. Figures 13 and 18 show, respectively, the paths taken by the robots for 9 time steps using the cooperative and the selfish control methods, and the change in the total scan certainty in time.

Based on Fig. 13 with the selfish controller, both robots prioritise victim v_1 over victim v_2, since these robots are steered by local controllers that follow (7)–(10), which prioritise visiting a cell that includes a victim with the worst health state and that is closer to the SaR robot (where the importance of each factor depends on the values for parameters c_1 and c_2). Since the cell that embeds victim v_1 meets both conditions, victim v_1 is the target of

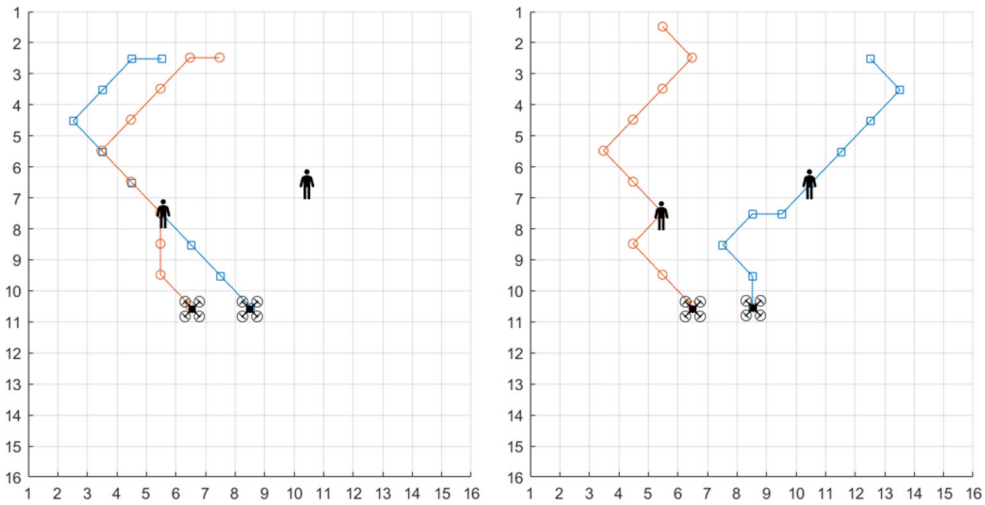

Fig. 13 Case 1 - Path taken by the SaR robots using the selfish controller (left) and the cooperative controller (right)

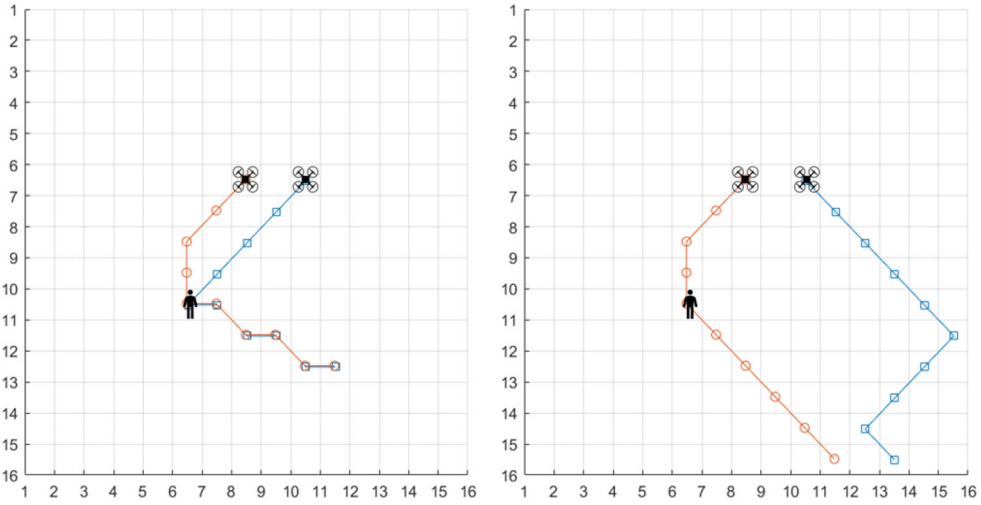

Fig. 14 Case 2 - Path taken by the SaR robots using the selfish controller (left) and the cooperative controller (right)

both SaR robots and is detected by them at time step 3. Afterwards, the robots continue exploring the environment without moving to victim v_2, who remains outside of their perception fields.

With the cooperative controller, however, SaR robot a_2 detects victim v_1 at time step 3 and SaR robot a_1 detects victim v_2 at time step 5 (see Figs. 8 and 13). This shows that the supervisory MPC controller has successfully coordinated the actions of the SaR robots in favour of detecting more victims within a given time span. More specifically, the second term in the objective function of the MPC controller (see Eq. 11) prevents the two SaR robots to cover the same sub-area of the environment. Since the local loss (considered by the first term in Eq. 11) for redirecting SaR robot a_1 towards victim v_2 is less than that of redirecting the other robot, the supervisory MPC controller changes the path that has been proposed by the local controller of SaR robot a_1 in order to achieve a higher global gain considering the scan certainty of the environment. The overall scan certainty for the cooperative controller is 17.2% larger than that of the selfish controller (see Fig. 18).

Case 2. Conflict in the Victim Detection and Area Coverage Consider 2 SaR robots and 1 victim in an environment of size 15×15 that is known except for sub-area $E' \subset E$ that is completely unknown, i.e., $c(x, y, 0) = 0$ for all $(x, y) \in E'$ and $c(x, y, 0) = 1$ for all $(x, y) \in E \setminus E'$ (see Fig. 9). Figures 14 and 19 show, respectively, the paths taken by the SaR robots for 9 time steps using the cooperative and selfish controllers and the change in the total scan certainty in time.

Figure 14 shows that for the selfish controller, both robots move towards victim v_1 and visit the victim at time step 4. Afterwards, both robots follow identical paths to approach the unknown sub-area. These individual behaviours are steered by the local controllers that, according to Eqs. 7-10, enforce each robot to prioritise cells that may embed a victim, are closer to the robot, and gain a higher percentage for the overall scan certainty of the environment. Since the cell that embeds the victim, in addition is closer to the robots

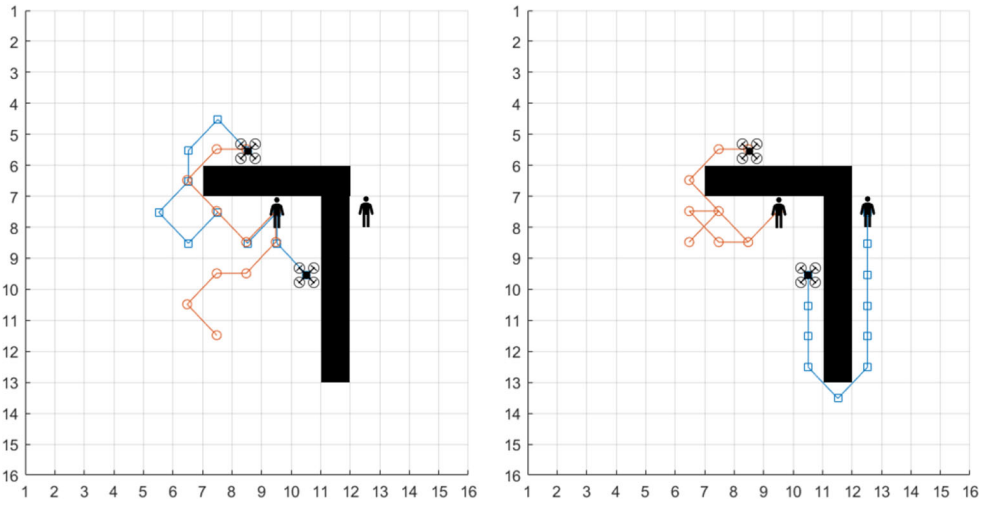

Fig. 15 Case 3 - Path taken by the SaR robots using the selfish controller (left) and the cooperative controller (right)

than the cells within sub-area E', this cell for the local controllers has a higher priority. After visiting the victim, based on Eq. 7, visiting the closest cell within sub-area E' becomes the priority of the local controllers.

With the cooperative controller SaR robot a_1 that is farther from the victim is redirected via the supervisory MPC controller to move directly towards sub-area E', while SaR robot a_2 moves towards victim v_1 and visits the victim at time step 4. With both the selfish and the cooperative controller, victim v_1 is detected equally fast, while based on Fig. 19, with the cooperative controller the overall scan certainty is almost 4 times larger than the value for the selfish controller. This is an influence of including the second term of Eq. 11 in the objective function of the supervisory MPC controller.

Case 3. Exploitation of the Perception Fields Consider 2 SaR robots and 2 victims in a partially known environment of size 15×15 with $c(x, y, 0) = 0.5$ for all $(x, y) \in E$ and a set of obstacles shown in black in Fig. 10. SaR robot a_1 has a sufficiently large perception field such that it detects both victims, whereas SaR robot a_2 detects victim v_1 only. The health states of victims v_1 and v_2 are, respectively, 20 and 15. Note that it is assumed that although the cells that include obstacles (shown in black) block the movement of the SaR robot, but they do not obstruct the view of the robot. Figures 15 and 20 show, respectively, the paths taken by the robots for 10 time steps using the cooperative and selfish controllers and the change in the total scan certainty in time.

With the selfish controller, both SaR robots visit victim v_1, which occurs at time steps 2 and 5 for robots a_1 and a_2, respectively (see Fig. 15). Although the health state of victim v_1 is less critical compared to that of victim v_2, victim v_1 is prioritised over victim v_2 by the local controller of SaR robot a_2, because this victim can be reached faster and thus the corresponding path receives a larger grade using Eq. 10. Afterwards, the SaR robots continue exploring the environment without detecting victim v_2 due to their limited perception fields.

With the cooperative controller, SaR robots a_1 and a_2 find victims v_2 and v_1 at time steps 10 and 5, respectively. Although SaR robot a_1 is able to reach victim v_1 in a shorter time, the global decision of the supervisory controller allows victim v_1 to be detected later in order to make sure that victim v_2 is detected in time (this is taken care of via the first term in Eq. 11, which considers the global gain of the path grades, instead of the individual/local ones). Thus this simulation highlights the ability of the cooperative controller to determine locally sub-optimal tasks for the SaR robots, in order to maximise the global mission performance. Figure 20 shows that by spreading out the SaR robots over the environment, the cooperative controller achieves an overall scan certainty that is 18.7% larger than that of the selfish controller.

Case 4: Exploitation of the Sensor Accuracies Consider 2 SaR robots in an environment of size 15×15, where the robots should scan a partially known environment with two sub-areas $E'_1, E'_2 \subset E$ (see Fig. 11). The scan certainty at the initial time step for sub-areas E'_1 and E'_2 is 0 and 0.3, respectively, and for all cells of E outside these two sub-areas is 0.9. Figures 16 and 21 show, respectively, the paths taken by the robots for 10 time steps using both the cooperative and the selfish controller, and the change in the total scan certainty in time.

Based on Fig. 16, with the selfish controller both SaR robots move to sub-area E'_1 to yield the highest gain in the scan certainty. With the cooperative controller, however, the robots move to sub-areas E'_1 and E'_2. Since SaR robot a_1 is closer to sub-area E'_1 and has a higher sensor accuracy, it is sent to sub-area E'_1 by the cooperative controller to yield a larger overall scan certainty. Based on Fig. 21 the overall

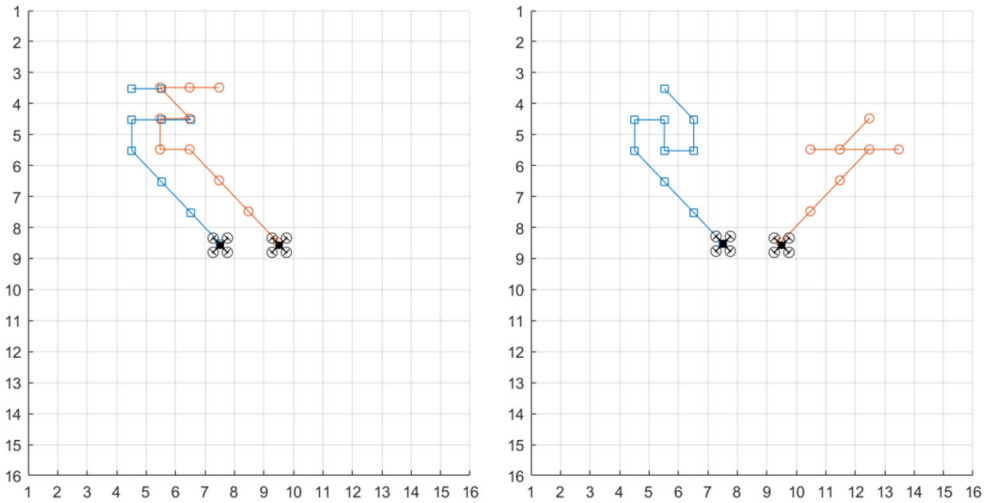

Fig. 16 Case 4 - Path taken by the SaR robots using the selfish controller (left) and the cooperative controller (right)

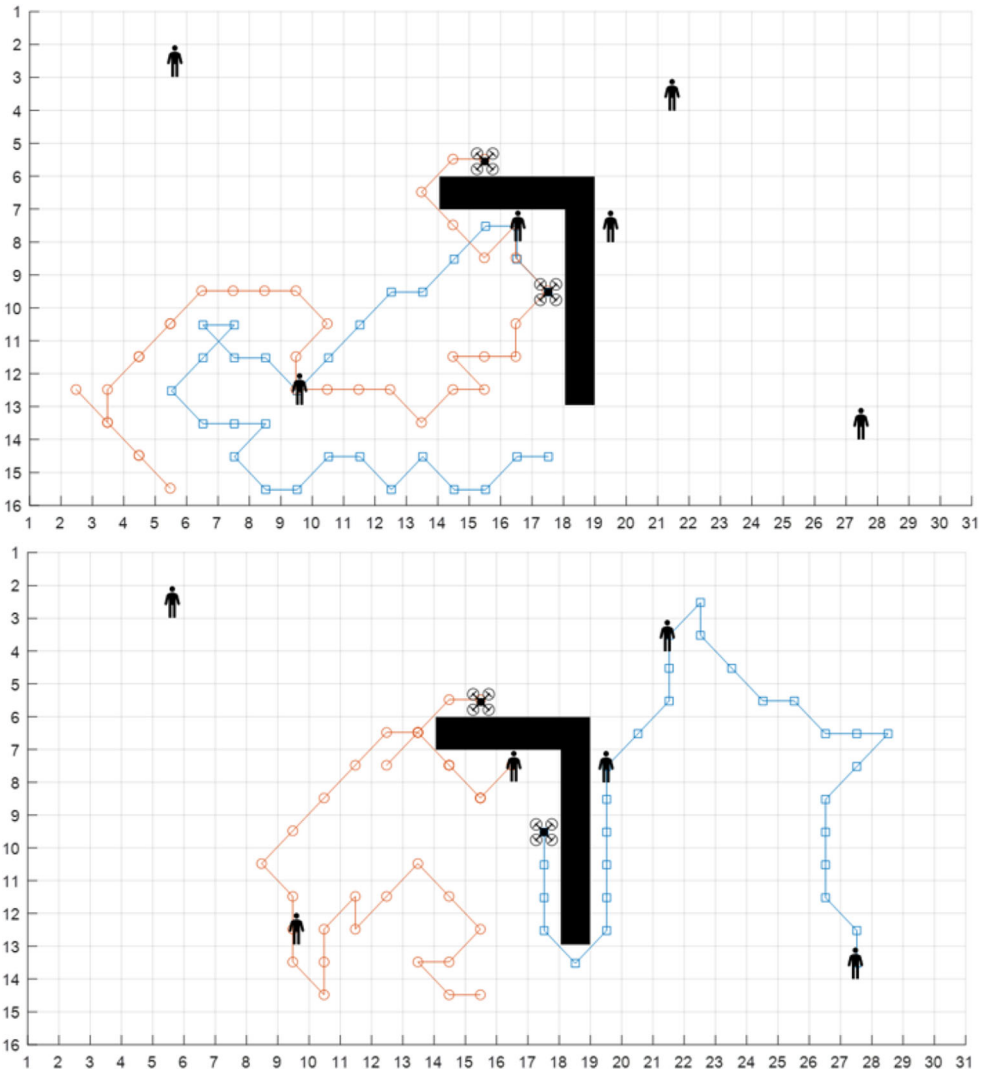

Fig. 17 Case 5 - Path taken by the SaR robots using the selfish controller (top) and the cooperative controller (bottom)

scan certainty of the cooperative controller is 29.8% larger than that of the selfish controller.

Case 5. Combined Scenario Consider 2 SaR robots, 6 victims, and a set of obstacles in an environment of size 30 × 15, where the robots should scan a partially known environment with 7 sub-areas $E'_1, \ldots, E'_7 \subset E$ (see Fig. 12). The scan certainties at the initial time step for sub-area E'_1 is 0.7, for sub-area E'_2 is 0.4, for sub-areas E'_3, E'_4, E'_5, E'_6 is 0.2, for sub-area E'_7 is 0.1, and for all cells of E outside these seven sub-areas is 0.5. The paths determined for both SaR robots via the selfish and cooperative controllers for 35 time steps are shown in Fig. 17. Additionally, Table 6 shows the health state of the

Table 6 Victim detection results for the combined scenario (case 5)

Selfish controller				Cooperative controller			
Victim	Health state	Number of visits	Detection time step	Victim	Health state	Number of visits	Detection time step
$\mathbf{v_1}$	6.94	0	–	$\mathbf{v_1}$	6.94	0	–
$\mathbf{v_2}$	24.9	2	10	$\mathbf{v_2}$	17.2	1	22
$\mathbf{V_3}$	17.97	2	3	$\mathbf{V_3}$	14.34	2	8
$\mathbf{V_4}$	0	0	–	$\mathbf{V_4}$	6.03	1	11
$\mathbf{V_5}$	6.94	0	–	$\mathbf{V_5}$	21.87	1	15
$\mathbf{V_6}$	6.94	0	–	$\mathbf{V_6}$	11.15	1	30

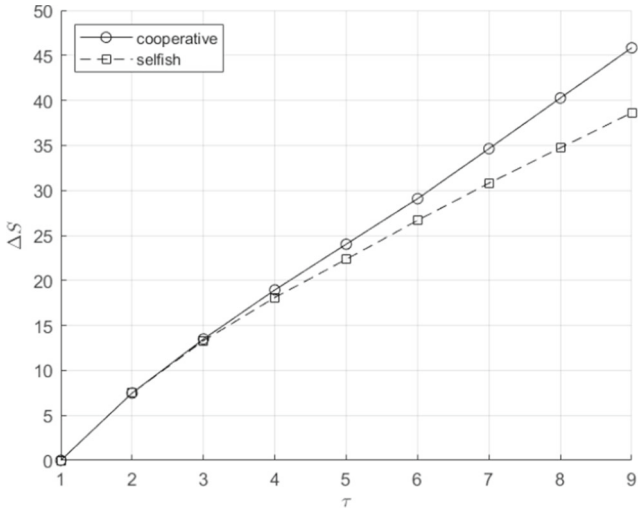

Fig. 18 Case 1 - Evolution of the scan certainty in time

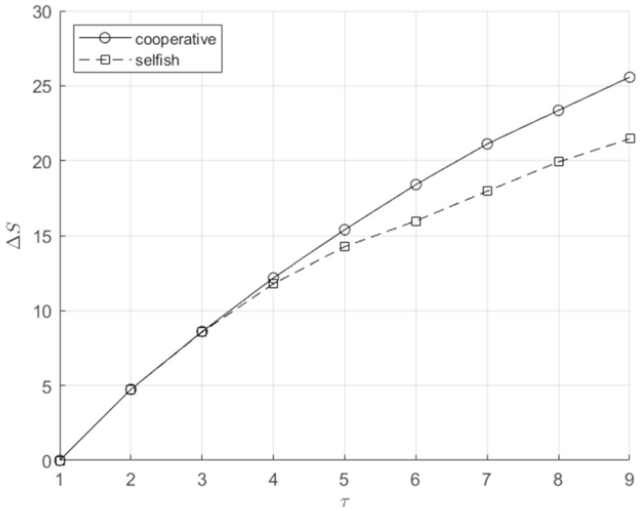

Fig. 20 Case 3 - Evolution of the scan certainty in time

victims at the end of the simulation, the number of times a victim has been visited, and the time step when each victim was first detected. The change in the total scan certainty in time is also illustrated in Fig. 22.

Figure 17 shows that with the selfish controller, both SaR robots visit victim v_3, and then move towards the southwest quadrant of the SaR environment, where they individually visit victim v_2. While 2 victims are visited doubly by the robots, 4 victims remain undetected and 1 victim deceases. With the cooperative controller, SaR robots a_1 and a_2 visit victims v_4 and v_3, respectively. Next they explore different sub-areas of the environment and detect additional victims v_2, v_5, and v_6. At the end only 1 victim remains undetected and no victim is deceased (Figs. 18, 19, 20 and 21).

The SaR system detects more victims with the cooperative controller, and no victim is visited twice, implying the victim search efficiency. Moreover, the overall scan certainty for the cooperative controller (see Fig. 22) is 27.6% larger than that for the selfish controller.

5 Conclusions and Topics for Future Research

Autonomous multi-robot systems are expected to map unknown search-and-rescue (SaR) environments in a fast and effective way. We have introduced a novel approach for coordinated mission planning of multi-robot systems for multi-objective (combined coverage and target-oriented) SaR. The proposed control approach effectively incorporates non-homogeneous imperfect perception capabilities of the sensors of different robots in order to improve

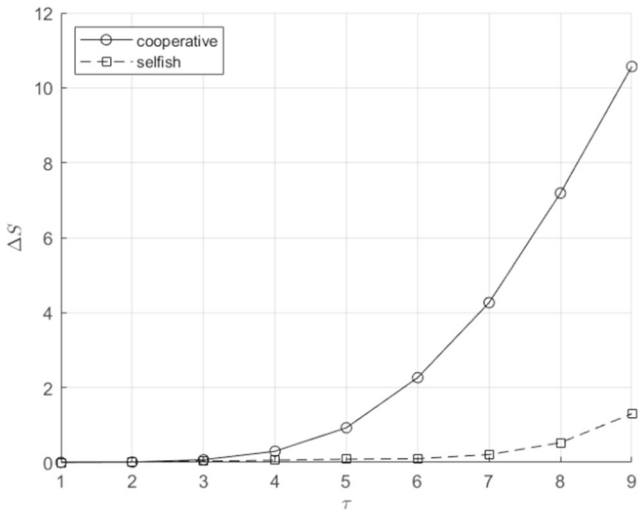

Fig. 19 Case 2 - Evolution of the scan certainty in time

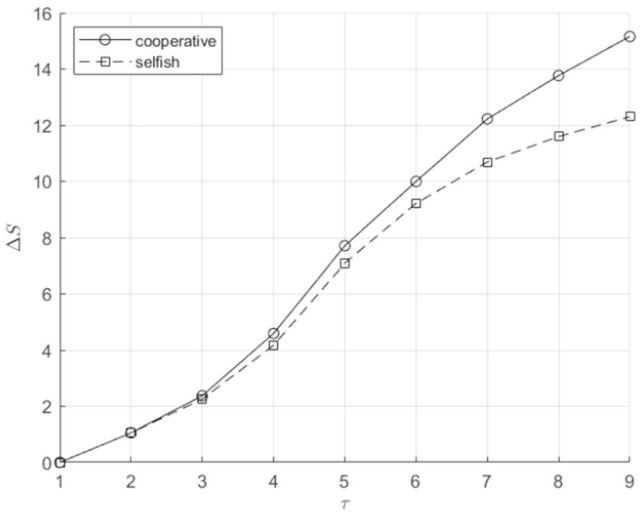

Fig. 21 Case 4 - Evolution of the scan certainty in time

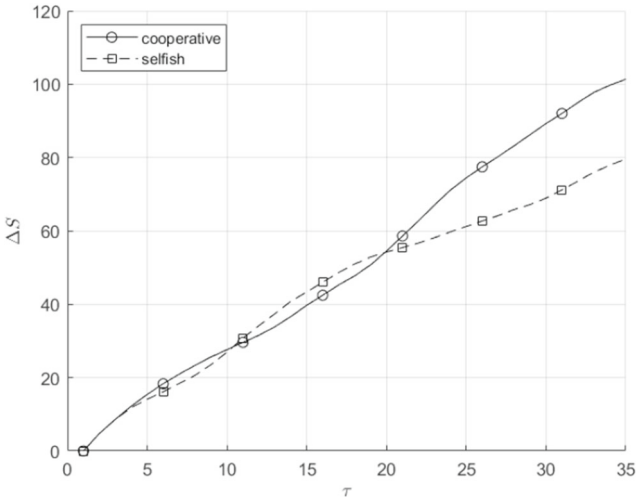

Fig. 22 Case 5 - Evolution of the scan certainty in time

their performance with respect to the victim detection and area coverage.

The key contributions of the paper are two-fold: in multi-agent control systems and in search-and-rescue (SaR) robotics. From the point-of-view of multi-agent control systems, we propose a novel control architecture and formulation that exploit the imperfect perception capabilities of agents, coordinate their decisions, and provide a balanced trade-off among various (conflicting) control objectives in a computationally efficient way. As it is also supported by our simulation results, the developed control system benefits from both the computational efficiency of decentralised control methods and the global vision of centralised control approaches. Additionally, the supervisory level improves the global control performance based on a predictive and optimal computation scheme, while local controllers independently steer the agents. Therefore, although the performance will expectedly degrade without the supervisory control level, the functioning of the multi-agent control system is robust to the failure of this centralised controller. Furthermore, the integrated formulation proposed in this research allows to incorporate both expert knowledge (via the fuzzy logic control systems) and the optimality and predictive capabilities of model predictive control (MPC) into the decision making of autonomous robots. These contributions are significant for SaR applications, because existing control methods are mainly focused on either coverage or target-oriented SaR. Moreover, MPC, which is a precise control method that systematically handles state and input constraints and that can provide robustness to SaR uncertainties, has been ignored in literature for the crucial task of area coverage in SaR. Our novel approach and formulation for multi-agent control systems enable MPC to provide all its strong points for, not only target-oriented, but also coverage-oriented SaR.

We have compared the performance of the resulting hierarchical control system with those of a decentralised selfish control system that excludes the MPC controller, a pure MPC controller, an ant-colony-based controller, and an exhaustive random search controller. In 20 simulated scenarios with randomly positioned obstacles and victims, the proposed hierarchical control approach showed the best performance in terms of the victim detection efficiency and the area coverage. Moreover, 5 structured scenarios were designed to simulate conflicting scenarios and to illustrate the importance of the proposed mathematical formulations in application. The results proved that in case of conflicts, with a comparable computation time, the proposed hierarchical controller significantly outperforms the decentralised controller. Moreover, the hierarchical controller successfully exploits the non-homogeneous perception capabilities of the robots, which improves the overall performance.

In the future, more detailed models that consider the behaviour, physical capabilities, and intentions of victims for their movement patterns can be considered. Furthermore, a systematic discussion and evaluation of the robustness of the proposed control approaches with respect to various sources of uncertainties, especially uncertainties in the movement of the victims, is a topic of interest for future research. Moreover, in addition to non-homogeneous perception capabilities, differences in the speed, degrees of freedom, computational capacity, tasks, and maneuverability of search-and-rescue robots can be considered. Additionally, combining autonomous learning methods within the proposed architecture is an interesting topic for future research. While using a learning-based approach alone may correspond to some risks for search-and-rescue applications, including such algorithms in a combined framework, similar to the one proposed in this paper, can result in a promising performance with adaptability capabilities. Finally, in real-life implementations the large size of SaR environments increases the computational burden of the supervisory MPC controller. To address this issue and also to mitigate the risk of the performance degradation due to the failure of the supervisory control level, a similar control architecture with more levels of control may be proposed. Thus, between the supervisory control level and the steering local control level, extra levels of control with several distributed MPC controllers may be considered, where each MPC controller supervises a combination of the local sub-areas.

Author Contributions Author C. de Koning contributed to designing and implementing the experiments. Authors C. de Koning and A. Jamshidnejad contributed to the analysis and interpretation of the results, development of the theoretical contributions, and composition of the manuscript. Author C. de Koning prepared the first draft of the manuscript. Author A. Jamshidnejad supervised the study design, and has reviewed, edited, and prepared the final version of the paper. Supervision and management of the project, as well as the funding

acquisition have been done by A. Jamshidnejad. Both authors have approved the final version of the manuscript.

Funding This research has been jointly supported by the NWO Talent Programme Veni project "Autonomous drones flocking for search-and-rescue" (18120), which has been financed by the Netherlands Organisation for Scientific Research (NWO), and by the TU Delft AI Labs & Talent programme.

Data Availability The data points, files, and codes for creating the figures represented in the results of this article are available online at https://figshare.com/s/9762330a7473363433ab.

Declarations

Competing interests The authors declare that they have no financial or non-financial conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit ¡ExternalRef¿ ¡RefSource¿http://creativecommons.org/licenses/by/4.0/¡/RefSource¿ ¡RefTarget TargetType="URL" Address="http://creativecommons.org/licenses/by/4.0/"/¿ ¡/ExternalRef¿.

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 Trans. Syst. Man Cybern. 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 10th World Conference on Earthquake Engineering, vol. 10, pp. 5989–5994, Rotterdam, Netherlands (1992)
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, Los Angeles, CA (2004)
4. Shimanski, C.: Situational awareness in search and rescue operations. In: International Technical Rescue Symposium (2005)
5. Chandarana, M., Hughes, D., Lewis, M., Sycara, K., Scherer, S.: Planning and monitoring multi-job type swarm search and service missions. J. Intell. Robot. Syst. **101**(44), 1–14 (2021)
6. Hong, A., Igharoro, O., Liu, Y., Niroui, F., Nejat, G., Benhabib, B.: Investigating human-robot teams for learning-based semi-autonomous control in urban search and rescue environments. J. Intell. Robot. Syst. **94**, 669–686 (2019)
7. Jamshidnejad, A., Frazzoli, E.: Adaptive optimal receding-horizon robot navigation via short-term policy development. In: 15th International Conference on Control, Automation, Robotics and Vision, pp. 21–28. IEEE (2018)
8. 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, Singapore (2016)
9. de Alcantara Andrade, F.A., Hovenburg, A.R., de Lima, L.N., Rodin, C.D., Johansen, T.A., Storvold, R., Correia, C.A.M., Haddad, D.B.: Autonomous unmanned aerial vehicles in search and rescue missions using real-time cooperative model predictive control. Sensors **19**(19), 4067 (2019)
10. San Juan, V., Santos, M., Andújar, J.M.: Intelligent UAV map generation and discrete path planning for search and rescue operations. Complexity **2018** (2018)
11. Yao, P., Zhao, Z.: Improved Glasius bio-inspired neural network for target search by multi-agents. Inform. Sci. **568**, 40–53 (2021)
12. Galceran, E., Carreras, M.: A survey on coverage path planning for robotics. Robot. Auton. Syst. **61**(12), 1258–1276 (2013)
13. Koenig, S., Liu, Y.: Terrain coverage with ant robots: a simulation study. In: Proceedings of the 5th International Conference on Autonomous Agents, pp. 600–607, Montreal, Canada (2001)
14. Wagner, I.A., Altshuler, Y., Yanovski, V., Bruckstein, A.M.: Cooperative cleaners: A study in ant robotics. Int. J. Robot. Res. **27**(1), 127–151 (2008)
15. Yang, S.X., Luo, C.: A neural network approach to complete coverage path planning. IEEE Trans. Syst. Man Cybern. B (Cybernetics) **34**(1), 718–724 (2004)
16. Yang, Y., Polycarpou, M.M., Minai, A.A.: Multi-UAV cooperative search using an opportunistic learning method. J. Dyn. Syst. Measu. Control **129** (2007)
17. Tutsoy, O., Barkana, D.E., Balikci, K.: A novel exploration-exploitation-based adaptive law for intelligent model-free control approaches. IEEE Trans. Cybern., 1–9. https://doi.org/10.1109/TCYB.2021.3091680 (2021)
18. Tutsoy, O., Barkana, D.E., Colak, S.: Learning to balance a Nao robot using reinforcement learning with symbolic inverse kinematic. Trans. Inst. Meas. Control. **39**(11), 1735–1748 (2017)
19. 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, Victoria, BC, Canada (2020)
20. Farrokhsiar, M., Pavlik, G., Najjaran, H.: An integrated robust probing motion planning and control scheme: A tube-based MPC approach. Robot. Auton. Syst. **61**(12), 1379–1391 (2013)
21. Hoy, M., Matveev, A.S., Savkin, A.V.: Collision free cooperative navigation of multiple wheeled robots in unknown cluttered environments. Robot. Auton. Syst. **60**(10), 1253–1266 (2012)
22. Cooper, J.R.: Optimal multi-agent search and rescue using potential field theory. In: AIAA Scitech 2020 Forum, p. 0879 (2020)
23. Paez, D., Romero, J.P., Noriega, B., Cardona, G.A., Calderon, J.M.: Distributed particle swarm optimization for multi-robot system in search and rescue operations. IFAC-PapersOnLine **54**(4), 1–6 (2021)
24. Choi, H., Brunet, L., How, J.P.: Consensus-based decentralized auctions for robust task allocation. IEEE Trans. Robot. **25**(4), 912–926 (2009)
25. Liu, Y., Nejat, G.: Robotic urban search and rescue: A survey from the control perspective. J. Intell. Robot. Syst. **72**(2), 147–165 (2013)
26. Best, G., Hollinger, G.A.: Decentralised self-organising maps for multi-robot information gathering. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4790–4797. IEEE/RSJ, Las Vegas (2020)
27. Otte, M., Kuhlman, M.J., Sofge, D.: Auctions for multi-robot task allocation in communication limited environments. Auton. Robot. **44**, 547–584 (2020)

28. Kashino, Z., Nejat, G., Benhabib, B.: Aerial wilderness search and rescue with ground support. J. Intell. Robot. Syst. **99**, 147–163 (2020)
29. Tol, D., Hoekstra, J., Jamshidnejad, A.: A bi-level local and global model predictive control architecture for air traffic management. In: 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), pp. 361–365 (2021)
30. Khamis, A.M., Elmogy, A.M., Karray, F.O.: Complex task allocation in mobile surveillance systems. J. Intell. Robot. Syst. **64**(1), 33–55 (2011)
31. 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, Pasadena, CA, USA (2008)
32. Chandler, P.R., Pachter, M., Rasmussen, S.: UAV cooperative control. In: Proceedings of the American Control Conference, vol. 1, pp. 50–55, Arlington, VA, USA (2001)
33. 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, Hull, UK (2018)
34. Sampedro, C., Rodriguez-Ramos, A., Bavle, H., Carrio, A., de la Puente, P., Campoy, P.: A fully-autonomous aerial robot for search and rescue applications in indoor environments using learning-based techniques. J. Intell. Robot. Syst. **95**, 601–627 (2019)
35. Krzysiak, R., Butail, S.: Information-based control of robots in search-and-rescue missions with human prior knowledge. IEEE Trans. Human-Mach. Syst. **52**(1), 52–63 (2021)
36. 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, Kollam, India (2011)
37. 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, Singapore (2013)
38. Dousai, N.M.K., Lončarić, S.: Detecting humans in search and rescue operations based on ensemble learning. IEEE Access **10**, 26481–26492 (2022)
39. Llasag, R., Marcillo, D., Grilo, C., Silva, C.: Human detection for search and rescue applications with UAVs and mixed reality interfaces. In: 2019 14th Iberian Conference on Information Systems and Technologies (CISTI), pp. 1–6 (2019)
40. Pinheiro, G.P.M., Miranda, R.K., Praciano, B.J.G., Santos, G.A., Mendonça, F.L.L., Javidi, E., da Costa, J.P.J., de Sousa, R.T.J.: Multi-sensor wearable health device framework for real-time monitoring of elderly patients using a mobile application and high-resolution parameter estimation. Frontiers in Human Neuroscience (2022)
41. Hart, P.E., Nilsson, N.J., Raphael, B.: A formal basis for the heuristic determination of minimum cost paths. IEEE Trans. Syst. Sci. Cybern. **4**(2), 100–107 (1968)
42. Yen, J.Y.: Finding the k shortest loopless paths in a network. Manag. Sci. **17**(11), 712–716 (1971)
43. Jamshidnejad, A., Papamichail, I., Papageorgiou, M., De Schutter, B.: Sustainable model-predictive control in urban traffic networks: Efficient solution based on general smoothening methods. IEEE Trans. Control Syst. Technol. **26**(3), 813–827 (2018)
44. Diehl, M., Bock, H.G., Schlöder, J.P.: A real-time iteration scheme for nonlinear optimization in optimal feedback control. SIAM J. Control. Optim. **43**(5), 1714–1736 (2005)
45. Kreinovich, V., Kosheleva, O., Shahbazova, S.N.: Why triangular and trapezoid membership functions: A simple explanation. In: Shahbazova, S.N., Sugeno, M., Kacprzyk, J. (eds.) Recent Developments in Fuzzy Logic and Fuzzy Sets: Dedicated to Lotfi A. Zadeh, pp. 25–31. Springer (2020)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Christopher de Koning received his BSc and MSc degrees in Aerospace Engineering from Delft University of Technology, The Netherlands, in 2018 and 2022, respectively. His research interests include fuzzy logic control, multi-agent robots, and search-and-rescue robotics.

Anahita Jamshidnejad received her PhD cum laude degree from Delft University of Technology, The Netherlands, in 2017. She is currently an Assistant Professor of Mathematical Decision Making at the Department of Control and Operations and the founder and co-director of the AI*MAN lab at Delft University of Technology. Her main research interests include optimization theory in engineering problems, AI-based control, model predictive, with applications to autonomous search-and-rescue and social robots.