A Novel MPC Formulation for Dynamic Target Tracking with Increased Area Coverage for Search-and-Rescue Robots

Baglioni, M.; Jamshidnejad, A.

DOI
10.1007/s10846-024-02167-3

Publication date
2024

Document Version
Final published version

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

Citation (APA)
Baglioni, M., & Jamshidnejad, A. (2024). A Novel MPC Formulation for Dynamic Target Tracking with Increased Area Coverage for Search-and-Rescue Robots. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *110*(4), Article 140. https://doi.org/10.1007/s10846-024-02167-3

Journal of Intelligent & Robotic Systems (2024) 110:140
https://doi.org/10.1007/s10846-024-02167-3

REGULAR PAPER

A Novel MPC Formulation for Dynamic Target Tracking with Increased Area Coverage for Search-and-Rescue Robots

Mirko Baglioni[1] · **Anahita Jamshidnejad**[1]

Received: 19 March 2023 / Accepted: 20 August 2024

Abstract
Robots are increasingly deployed for search-and-rescue (SaR), in order to speed up rescuing the victims in the aftermath of disasters. These robots require effective mission planning approaches to determine time and space-efficient trajectories that steer them faster towards (moving) victims, while dealing with uncertainties. Model predictive control (MPC) is an effective optimization-based control approach that has been used to steer robots along reference trajectories determined by higher level controllers. Determining the trajectory of the robots directly via MPC has the advantage of optimizing multiple SaR criteria while handling the constraints. We, thus, introduce a path planning approach based on MPC for indoor SaR robots that allows the robot to systematically chase the moving victims, when no reference trajectory is provided. The proposed approach combines target-oriented and coverage-oriented search, and allows for systematic handling of environmental uncertainties, by deploying a robust tube-based version of the introduced MPC formulation. In addition, we model the movements of the victims for MPC, by adopting an existing evacuation model. We present a case study, using Gazebo, MATLAB, and ROS, where the performance of the proposed MPC controller is evaluated compared to four state-of-the-art methods (two target-oriented methods based on MPC and A* and two heuristic algorithms for area coverage). The results show that, while robust to uncertainties, our approach overall outperforms the other methods, with regards to victim detection, area coverage, and mission time.

Keywords Model predictive control · Robot path planning · Target-oriented and coverage-oriented search-and-rescue

1 Introduction

Recently search-and-rescue (SaR) robots are increasingly used in dangerous and complicated stages of SaR [1–4]. Robots are expendable and can access locations that are inaccessible to humans. By taking over dangerous tasks, robots allow humans to contribute to other tasks, e.g., providing medical and emotional support for victims [5, 6]. Novel control approaches are needed to enable robots to autonomously and time-efficiently search for trapped people [7].

Model predictive control (MPC) [8] is an optimization-based control approach that explicitly incorporates various state and input constraints, and finds balanced trade-offs between various objectives of SaR missions. Furthermore, robust versions of MPC have been established (see, e.g., [9–11]) that, if adopted for SaR robots, will maintain their performance in presence of uncertainties [12–15]. While a well-known challenge of MPC methods is the demanding computations, a vast number of literature exists that propose alternative approximate MPC methods to tackle this challenge (see, e.g., [16, 17]).

Although MPC has been used for SaR robotics, the applications mainly concern tracking a reference trajectory that is determined by another controller [18, 19]. However, exploiting the advantages of MPC for systematic exploration of SaR environments when no reference trajectory is available or when it is desired to determine this trajectory directly via MPC remains limited (see Section 1.1). Additionally, reference-tracking MPC is used for target-oriented SaR, whereas optimizing the area coverage, next to moving towards the target victims, possesses two main advantages: First, the estimated positions for the target victims are usually prone to uncertainties. Therefore, a solely reference point tracking method may lead the robot to places without victims and thus, the robot should include exploration to its track-

✉ Mirko Baglioni
M.Baglioni@tudelft.nl

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

Published online: 24 September 2024

ing task. Second, area coverage is essential for various SaR missions, in order to find unknown victims or other important targets (e.g., explosives). Running the area coverage and target tracking as two separate or sequential missions will pose additional challenges regarding real-time communication and data exchange and coordinating the robots, and will result in increased mission time, over-populating the area with robots, etc. Designing an MPC system that determines on the go, according to a constrained (multi-objective) optimization problem, which of the two tasks the robot should take will eliminate those issues.

In this paper, we propose a novel formulation based on MPC for autonomous decision making of SaR robots, when no reference trajectories exist and the path should be planned by MPC itself. Our main motivation for determining the trajectory of the robots directly via MPC is to optimize the SaR mission with respect to time and the area coverage, especially the coverage of areas that potentially include victims, and to systematically handle the hard constraints via SaR robots. The area coverage problem is dynamic, due to the possibility of movement of the victims or obstacles. Moreover, our main aim is to adopt MPC for systematic combined target and coverage-oriented SaR, while handling various state and control constraints. Satisfying constraints, e.g., avoiding obstacles, fire, and collision, reaching specific terminal targets, and incorporating the limits of the actuators and the kinematics of the robots, is crucial for SaR robots.

The proposed generalized MPC-based framework can be adopted for mission planning of robots for various environments, targets, robots, and unmodeled disturbances in SaR missions. This framework will exploit the unique characteristics of MPC for mission planning of SaR robots. Moreover, various models for the SaR environment, dynamic targets (e.g., victims), and robots and their actuators can be plugged into MPC as prediction models, which significantly expands the applicability and generalizability of our proposed framework.

The main contributions of this paper include:

- A novel MPC-based formulation for mission planning of SaR robots in dynamic environments is introduced. While MPC for SaR robots has mainly been used for (robust) reference tracking, which leads the robot to specific targets (target-oriented SaR), we formulate an economic MPC problem that includes both target-oriented and coverage-oriented objectives and constraints. This will significantly improve the efficiency and effectiveness of search-and-rescue, without having a reference trajectory.
- We adopt a dynamic force-based evacuation model to represent the movements of the targets (victims). We also incorporate the uncertainties regarding the exact position of the victims and the evolution of these uncertainties into the model, which will be used by MPC as prediction model. The proposed control approach is not limited to this particular model and may be integrated with different dynamic models for victim/target movement.
- We design and run a case study (implemented in MATLAB, ROS, and Gazebo) to assess the performance of the proposed MPC method with respect to four different state-of-the-art methods, that are dedicated target-oriented or coverage-oriented approaches. For the case study, we adopt a robust tube-based version of the proposed MPC formulation, in order to deal with environmental uncertainties included in victims locations and in the robot model. We also include the results of real-life experiments in the lab.

The remainder of the paper is structured as the following. Section 1.1 provides a background discussion about the control of SaR robots for various SaR objectives. In Section 2, we describe the proposed methodologies, including the MPC formulation and the dynamic model for the movement of the victims. In Section 3, we show and discuss the results of a case study that compares our approach with four different state-of-the-art methods for simulated indoor SaR missions. Finally, Section 4 concludes the paper and proposes topics for future research. Moreover, Table 1 shows the commonly used mathematical notations.

1.1 Background

Considering the main objectives of their control systems, SaR missions may be categorized as *coverage-oriented* [20, 21] and *target-oriented* [13, 14]. In coverage-oriented SaR, exploring the SaR environment and finding the victims [22–25] is the focus of the control system, whereas in target-oriented SaR the control system aims at reaching a specific target (e.g., an exit) [26]. The most commonly used coverage-oriented approaches for mission planning of SaR robots include heuristics techniques (e.g., bug algorithms [27], potential fields methods [28], fuzzy logic control [29], and particle swarm optimization [30]). Graph-based [31, 32] and optimization-based control approaches, including MPC, are often used in target-oriented SaR. An extensive literature survey shows that the use of MPC in coverage-oriented SaR is rather limited (see, e.g., [33–37]). In particular, MPC is used for reference tracking [18], i.e., when a reference path that steers the robot to a specific known destination is available.

We first discuss some state-of-the-art coverage-oriented approaches. Although the focus of our paper is on single robot control, in order to properly cover various control methods, we consider papers with both single and multiple robot systems. Since most approaches decompose the environment into cells, a main classification of coverage-oriented methods includes exact and approximate decomposition methods.

Table 1 Table of frequently used mathematical notation

Variable	Description
MPC	
$A_i(k)$	Area that is expected to include victim i at time step k
$C^{\mathrm{p}}(k)$	Circular area representing the perception field of the SaR robot at time step k
k^{c}	Control time step
k^{s}	Simulation time step
$\mathcal{M}^{\mathrm{Evac}}$	Mapping corresponding to "Evac" crowd evacuation model
N^{p}	MPC prediction horizon
$\boldsymbol{r}^{\mathrm{rob}}(k)$	Position of the robot at time step k
T^{c}	Control sampling time
$v^{\mathrm{rob}}(k)$	Linear velocity of the robot at time step k
$\omega^{\mathrm{rob}}(k)$	Angular velocity of the robot at time step k
$x^{\mathrm{rob}}(k)$	Coordinate x of the robot at time step k
$y^{\mathrm{rob}}(k)$	Coordinate y of the robot at time step k
$\theta^{\mathrm{rob}}(k)$	Orientation angle of the robot at time step k
π^{c}	Threshold used by MPC for intersection of the areas $C^{\mathrm{p}}(k)$ and $A_i(k)$ to stop chasing victim i
Evacuation model	
$\boldsymbol{F}_i(k)$	Total external force on victim i at time step k
$\boldsymbol{F}^{\mathrm{s}}_{ij}(k)$	Social force between victims i and j at time step k
$\boldsymbol{F}^{\mathrm{c}}_{ij}(k)$	Contact force between victims i and j at time step k
$\boldsymbol{F}^{\mathrm{a}}_{ij}(k)$	Attraction force between victims i and j at time step k
$\boldsymbol{F}^{\mathrm{s}}_{io^{\mathrm{s}}}(k)$	Social force between victim i and static obstacle o^{s} at time step k
$\boldsymbol{F}^{\mathrm{c}}_{io^{\mathrm{s}}}(k)$	Contact force between victim i and static obstacle o^{s} at time step k
$\boldsymbol{F}^{\mathrm{a}}_{io^{\mathrm{d}}}(k)$	Attraction force between victim i and dynamic obstacle o^{d} at time step k
I_i	Moment of inertia of victim i
m_i	Mass of victim i
$\boldsymbol{r}^{\mathrm{v}}_i(k)$	Position of victim i at time step k
$T_i(k)$	Total external torque on victim i at time step k
$T^{\mathrm{s}}_i(k)$	Torque of the social force on victim i at time step k
$T^{\mathrm{c}}_i(k)$	Torque of the contact force on victim i at time step k
$T^{\mathrm{m}}_i(k)$	Torque of the motive force on victim i at time step k
$\boldsymbol{v}^0_i$	Velocity vector field of victim i
$\boldsymbol{v}^{\mathrm{v}}_i(k)$	Linear velocity of victim i at time step k
$\eta_i(k)$	Fluctuation torque component on victim i at time step k
$\theta^{\mathrm{v}}_i(k)$	Orientation angle of victim i at time step k
$\boldsymbol{\xi}_i(k)$	Fluctuation force component on victim i at time step k
τ_i	Time relaxation parameter of victim i

Using an exact decomposition, Agarwal and Akella in [38] consider a multi-robot system for exploring a given 2-dimensional area, where the robots have constraints on their flight time and battery life. The problem is transformed into a line-coverage one, which is then solved by a graph-based method. In [39], a coverage-oriented path planning method based on approximate cellular decomposition is proposed that is shown to reduce the coverage overlap. Their approach, however, does not account for dynamic obstacles and uncertainties. In [40] a bio-inspired ant colony optimization algorithm is proposed, where variable velocities are introduced for the ants, in order to find sub-optimal paths, while reducing the overall path planning time. This approach resulted in improved computational efficiency and reduced repetition in covering the same areas. In [41] four methods are proposed for discrete-space path planning of unmanned aerial vehicles in SaR missions. Using fuzzy logic, a map of the search area is generated where places with

a higher risk and a higher probability of finding the victims are specified. The four methods for path planning are based on artificial intelligence, and thus systematic obstacle avoidance and optimization of the control inputs have not been considered. In [42] a distributed particle swarm optimization algorithm for path planning of a multi-robot SaR system is proposed. The robots use artificial potential functions to avoid static obstacles in the environment and to reach the victims. With respect to classical particle swarm optimization methods, through the introduction of a repulsive force among the particles and robots, the performance is improved considering exploration of the area and avoiding collisions between the robots. However, uncertainties in the SaR environment and dealing with moving obstacles have not been considered in [42]. Other papers that have considered heuristic control of SaR robots include [43–47], which particularly focus on reinforcement learning. A main challenge with reinforcement-learning-based approaches, however, is that these methods do not systematically incorporate the constraints of a SaR mission. To summarize, optimality (as a balanced trade-off) with respect to multiple competing criteria and incorporation of constraints are not systematically handled by these approaches.

In [33] a control architecture including an MPC layer is presented for combined target-oriented and coverage-oriented mission planning of SaR robots. The architecture is suitable for multi-robot systems and the main task of MPC is coordination of the robots, rather than directly steering them based on the solutions of a constrained multi-objective optimization problem. The robots are steered by individual local heuristic fuzzy logic controller towards various targets. In summary, the common aspect of our paper and [33] is in adoption of MPC for systematic maximization of the coverage of (partially) unknown SaR environments. However, while in [33] MPC is a supervisory layer that only acts when local fuzzy logic controllers fail to cover the area above a given threshold, in our paper a reformulation of the objective function and constraints of MPC is proposed for direct steering of SaR robots. The resulting framework is generalizable, as explained earlier, and exploits the unique advantages of MPC for various SaR scenarios. Since the architecture in [33] follows the same objective as our paper, i.e., approaching more target victims, while maximizing the area coverage, we compare the performance of our approach with that architecture.

In [34] distributed MPC is used for discrete-space and discrete-time path planning of a cooperative multi-drone system that searches an outdoor environment. The objective of MPC is to determine the speeds and roll angles of the drones in order to maximize their reward for visiting the cells of an environment, which is prone to changes due to the wind flow. In [48], Carron and Zeilinger propose a coverage control method based on nonlinear MPC for a multi-robot system with nonlinear dynamics and state and input constraints. Moreover, a distributed coverage control problem is solved in [35] via MPC. The approaches in both papers converge to a centroidal Voronoi configuration. Our problem differs from that in these papers, since we seek a trade-off between systematic chasing of dynamic targets in uncertain environments (which is not considered in [48] and [35]) and maximizing the area coverage dynamically. In [36] and [37], Ibrahim et al. present control approaches for path planning of robots for area coverage, based on MPC. Compared to their bi-level discrete-space approach, we employ only one level of MPC for a continuous-space problem. The last two papers introduce dynamic obstacles within the environment that the robot should avoid. This is different from our problem where some dynamic victims are assigned as targets for a robot and the robot should estimate the future evolution of their movement, including possible uncertainties, and chase them time and space-efficiently.

Next, we give an overview on papers that consider target-oriented SaR. The number of papers in this category that use MPC for path-planning is much more vast compared to coverage-oriented SaR. In [49], a path planning approach based on decentralized robust MPC is proposed for cooperative, collision-free navigation of multiple wheeled robots in an unknown, static, and cluttered environment towards static targets with known positions. One MPC layer is used to determine way-points according to nominal trajectories for the robots, and a second MPC layer uses a robust approach to track these trajectories despite uncertainties. Farrokhsiar et al. develop a two-layer robust tube-based MPC controller in [50] for motion planning of a unicycle robot that should reach a static target with a known position in a cluttered environment. Their results show that the robust tube-based MPC controller stabilizes the position of the robot around the planned nominal trajectory in dynamic cluttered environments.

MPC is also used in [51] for mission planning of an autonomous ground robot in an unknown environment with static obstacles that should reach a static target with a known position, while maintaining a safe distance from the obstacles. The robot has access only to information within its perception field, and accordingly determines intermediary goal positions and moves towards them using an MPC controller with two possibly competing objectives, i.e., reduction of the mission time and the energy consumption. While the results show good performance and constraint satisfaction in an environment with static obstacles, dynamic obstacles and targets, and uncertainties in the position of the target(s) remain topics for future research. In [19] MPC is used in an outdoor mission with an unmanned aerial vehicle for tracking dynamic targets with known initial positions and speeds, considering the wind flow, in an obstacle-free environment.

In connection to the topic of our paper, we also consider

works on search and pursuit-evasion in mobile robotics [52]. In [53], a distributed algorithm for target tracking by a group of pursuers is presented, introducing a modified version of the Voronoi tassellation that generates collision-free areas. The pursuers are capable of tracking the dynamic target even with an uncertain position. A drawback of the approach is that the presence of a barricade of obstacles does not allow reaching the target. In this field, there are also works that use MPC. In fact, in [54] an MPC approach is proposed and compared to a game-theoretic method. It is shown that MPC can solve this problem with less information required and with a similar performance. In more detail, in [55] the same authors show that MPC requires less information to solve pursuer-evasion games, where in particular the position of the opponent is required, but its orientation and dynamics are not. In [56] De Simone et al. introduce a pursuer-evader problem for humanoids in presence of obstacles. Firstly, unicycle models are used to generate reference velocities, and lastly, MPC is used for tracking them for stable gait generation.

Finally, other related topics are robot exploration and target-driven navigation. The first group include studies that discuss and propose exploration approaches, many of which use learning-based techniques, that can be reinforcement learning [57] or deep learning [58]. In particular, in [59] deep reinforcement learning is combined with the more traditional approach of frontier-based exploration for autonomous exploration of unknown cluttered environments by a SaR robot, while in [60] an approach based on reinforcement learning is used that can leverage structural regularities of the environment, robustness to errors in state-estimation, and flexibility with respect to input modalities, or also in [61] a multi-robot exploration strategy in presence of communication dropouts is presented. The second group includes papers on navigation approaches guided by target information, where many use reinforcement learning [62, 63], foundation models based on deep learning [64, 65], or solve optimization problems [66]. In more detail, [62] proposes a navigation approach to reach a target only using vision information based on two networks that respectively explore the environment and locate the target, or in [63] an attention-based method is proposed to learn to navigate by leveraging an episodic memory that embeds previously-visited states, while in [67] a frontier-based exploration method that exploits visual information to navigate towards unseen semantic objects is presented.

2 Proposed Methodologies

In this section, we explain our proposed methodology, including the problem definition, assumptions and models, and mathematical formulations.

2.1 Problem Statement

The SaR control problem is formulated within a discrete-time and continuous-space framework. We consider an autonomous ground robot that follows the standard unicycle model (see Section 2.2.1 for details). This robot should explore a partially known 2-dimensional SaR environment $\mathcal{E}$, while chasing specific moving targets. Figure 1 illustrates an example of a SaR environment and its static and dynamic elements. More specifically, the SaR environment contains a time-invariant set $\mathcal{O}^{\mathrm{s}}$ of sub-areas occupied by static and closed obstacles, e.g., debris and stones with generally arbitrary shapes (see the dashed cyan shapes in Fig. 1; set $\mathcal{O}^{\mathrm{s}}$ also includes a safety margin for the robot around each obstacle), and a time-varying set $\mathcal{O}^{\mathrm{d}}(k)$ of sub-areas occupied by moving objects at time step k, e.g., victims ($\mathcal{O}^{\mathrm{d}}(k)$ includes a safe margin that allows the robot to get as close as needed to its target victims, without crashing into or disturbing them).

The assumptions used are given below:

A1 The robot is associated with a set $\mathcal{V}$ of N^{v} moving target victims and is initially given a set $\mathcal{A} = \{A_1(0), \ldots, A_{N^{\mathrm{v}}}(0)\}$ of N^{v} closed circular areas that

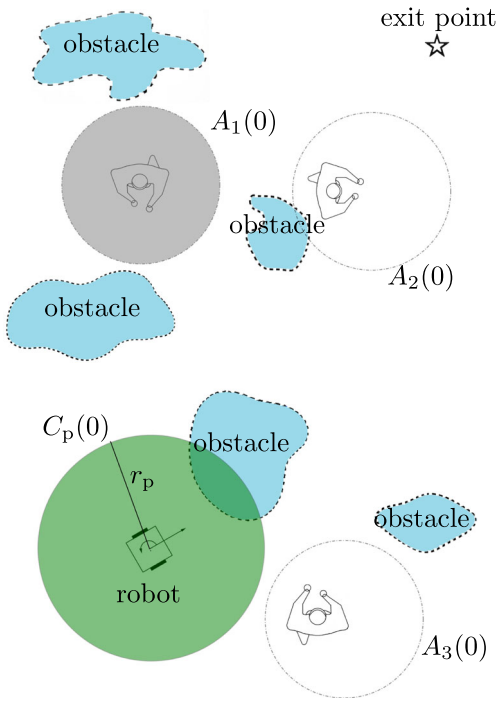

Fig. 1 Search-and-rescue environment at initial time step $k = 0$ including the SaR robot and its perception field $C^{\mathrm{p}}(0)$ (illustrated with a solid green circle), three victims and their approximate initial areas $A_1(0)$, $A_2(0)$, $A_3(0)$ that represent the uncertain locations of the victims (illustrated with dash-dotted grey circles in case of target victims, or with dash-dotted uncolored circles in case of non-target victims), static obstacles (illustrated with dashed cyan circles), and the exit (illustrated by a star symbol)

represent the approximate initial location of each target victim (i.e., $\boldsymbol{r}_i^{\mathrm{v}}(0) \in A_i(0)$, with $i = 1, \ldots, N^{\mathrm{v}}$). Victim i is assumed to initially be inside area $A_i(0)$. Before the robot detects a target victim, it only has knowledge about the approximate initial area $A_i(0)$ of the victim. Thus the robot predicts the evolution and transition of areas $A_i(k)$ in time according to a victims movement model.

A2 An initial exploration phase of the environment has already resulted in knowledge about the approximate initial location of the victims and the position, size, and number of the static obstacles. In real world, the bounded uncertainties corresponding to the areas where victims are located may be estimated based on the likely whereabouts of the victims (e.g., offices, canteens, bedrooms), depending on the functionality and expected number of inhabitants of the building, size and position of the rooms, time of the day.

A3 From every position in the environment, where the robot may be placed, there is a feasible path to the final target of the robot (i.e., the exit).

A4 The robot has perfect knowledge of its own states at every time step.

A5 The robot is equipped with a sensor with a circular perception area $C^{\mathrm{p}}(k)$ of fixed radius r^{p} (see the solid green circle in Fig. 1). The center of the perception field always coincides with the position $\left(x^{\mathrm{rob}}(k), y^{\mathrm{rob}}(k)\right)$ of the center of gravity of the robot. The sensor has perfect perception. At every time step, the sensor provides a map of the sub-area that is encountered by $C^{\mathrm{p}}(k)$, including the shape, size, and location of the perceived obstacles[1].

The robot uses a dynamic model to predict the evolution of the approximate areas $A_1(k), \ldots, A_{N^{\mathrm{v}}}(k)$ in time (see Section 2.2.2 for details), which may be influenced by a motive force field, social forces among the victims, and repulsion forces with respect to static obstacles.

The aim of this paper is to develop a dynamic and online mission planning control system for the robot that steers the robot within the free space $\mathcal{E} \setminus \left(\mathcal{O}^{\mathrm{s}} \cup \mathcal{O}^{\mathrm{d}}(k)\right)$ towards its final destination (an exit represented by a star symbol in Fig. 1) in the shortest possible time, while the robot maximizes a trade-off between its area coverage and the possibility of visiting more target victims from the set $\mathcal{V}$, continuously satisfying the hard state and input constraints. Therefore, the SaR problem should be formulated as a combined target-oriented and coverage-oriented multi-objective MPC problem that incorporates all the constraints (see Section 2.2.3 for details).

[1] Sensor fusion and robustness to sensor inaccuracies are out of the scope of this paper and are thus assumed to be provided for the robot.

2.2 MPC for Combined Dynamic Target Chasing and Area Coverage for SaR Robots

We first explain the models used for the mobile robot and for movement of the victims. Then we detail the MPC formulation.

2.2.1 Kinematics Model for the Ground Robot

For the robot, we consider the following kinematics model, which is a standard unicycle model:

$$x^{\mathrm{rob}}(k^{\mathrm{s}}+1) = x^{\mathrm{rob}}(k^{\mathrm{s}}) + v^{\mathrm{rob}}(k^{\mathrm{s}}) \cos\left(\theta^{\mathrm{rob}}(k^{\mathrm{s}})\right) T^{\mathrm{s}} + \delta_x(k^{\mathrm{s}}) \tag{1a}$$

$$y^{\mathrm{rob}}(k^{\mathrm{s}}+1) = y^{\mathrm{rob}}(k^{\mathrm{s}}) + v^{\mathrm{rob}}(k^{\mathrm{s}}) \sin\left(\theta^{\mathrm{rob}}(k^{\mathrm{s}})\right) T^{\mathrm{s}} + \delta_y(k^{\mathrm{s}}) \tag{1b}$$

where $[\boldsymbol{r}^{\mathrm{rob}\top}(k^{\mathrm{s}}), \theta^{\mathrm{rob}}(k^{\mathrm{s}})]^{\top}$ is the state vector of the robot at simulation time step k^{s}, with $\boldsymbol{r}^{\mathrm{rob}}(k^{\mathrm{s}}) = [x^{\mathrm{rob}}(k^{\mathrm{s}}), y^{\mathrm{rob}}(k^{\mathrm{s}})]^{\top}$ including the x and y coordinates corresponding to the center of gravity of the robot and the angular position of the robot with respect to the horizontal axis (see Fig. 2). Moreover, $v^{\mathrm{rob}}(k^{\mathrm{s}})$ is the linear velocity of the robot at simulation time step k^{s} and T^{s} is the simulation sampling time. The model mismatch with respect to the real robot is formulated via bounded uncertainties $\delta_x(k^{\mathrm{s}})$ and $\delta_y(k^{\mathrm{s}})$; these uncertainties can represent the non-smoothness of the terrain in a SaR environment, that makes the robot real position deviate from the nominal one. With this model, we assume that the linear and rotational motions of the robot are decoupled, i.e., at

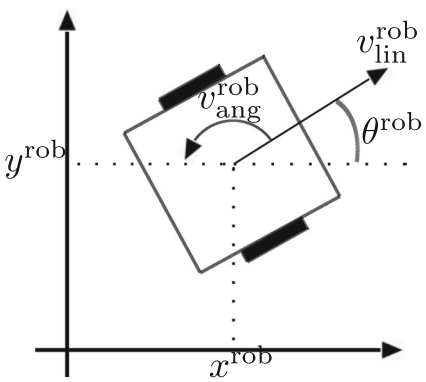

Fig. 2 Two-wheeled ground mobile robot schematic

the beginning of every simulation sampling time, the robot makes a rotation according to $\theta^{\text{rob}}(k)$ (with its maximum possible angular speed and in a negligible time) and continues its movement according to $v^{\text{rob}}(k^{\text{s}})$. Note that the linear velocity $v^{\text{rob}}(k^{\text{s}})$ and the angular position $\theta^{\text{rob}}(k^{\text{s}})$ are control inputs (i.e., optimization variables of the MPC).

2.2.2 Dynamic Model for Movement of the Victims

In order to model the movement of the victims and thus estimate the evolution of their approximate areas, we consider the crowd evacuation model FDS+Evac [68], which is a well established model used, e.g., in [69–71].

Remark *Our approach is independent of the selected crowd evacuation model, and thus a different model can be used. For this reason, the application is not limited to SaR. However, in order to properly use FDS+Evac model, in case moving obstacles exist in the SaR environment, the robot should be aware of them and should have a model of their motion.*

In FDS+Evac model, the movements including the position and orientation of humans in a disaster scene are driven by (physical and virtual) forces and torques, such as the contact forces and the gravity, as well as the psychological forces, based on the social force model by Helbing and Molnar [72], that are exerted by the environment. The movement trajectories of the victims in this model are formulated in continuous time and continuous space. The equations of motion for target victim i ($i = 1, \ldots, N^{\text{v}}$) are given by:

$$m_i \frac{\mathrm{d}^2 \boldsymbol{r}_i^{\text{v}}(t)}{\mathrm{d}t^2} = \boldsymbol{F}_i(t) + \boldsymbol{\xi}_i(t) \tag{2}$$

$$I_i \frac{\mathrm{d}^2 \theta_i^{\text{v}}(t)}{\mathrm{d}t^2} = T_i(t) + \eta_i(t) \tag{3}$$

where m_i is the mass of target victim i, $\boldsymbol{r}_i^{\text{v}}(t) = [x_i^{\text{v}}(t), y_i^{\text{v}}(t)]^\top$ is the position of the target victim at time instant t with $x_i^{\text{v}}(t)$ and $y_i^{\text{v}}(t)$ the x and y coordinates, $\boldsymbol{F}_i(t)$ is the total measured external force on target victim i at time instant t, $\boldsymbol{\xi}_i(t)$ is a random fluctuation force at time instance t, I_i is the moment of inertia of target victim i, $\theta_i^{\text{v}}(t)$ is the angular position (i.e., heading) of the target victim at time instant t, $T_i(t)$ is the total measured external torque on the target victim at time instant t, and $\eta_i(t)$ is a random fluctuation torque at time instant t. The fluctuation components $\boldsymbol{\xi}(t)$ and $\eta_i(t)$ are part of the uncertainties that affect the control system of the robot.

The total measured external force and torque on target victim i are given by:

$$\begin{aligned}\boldsymbol{F}_i(t) = {} & \frac{m_i}{\tau_i}(\boldsymbol{v}_i^0 - \boldsymbol{v}_i^{\text{v}}(t)) + \\ & \sum_{\substack{j=1 \\ j \neq i}}^{N^{\text{v}}} \left(\boldsymbol{F}_{ij}^{\text{s}}(t) + \boldsymbol{F}_{ij}^{\text{c}}(t) + \boldsymbol{F}_{ij}^{\text{a}}(t) \right) + \\ & \sum_{o^{\text{s}} \in \mathcal{O}^{\text{s}}} \left(\boldsymbol{F}_{io^{\text{s}}}^{\text{s}}(t) + \boldsymbol{F}_{io^{\text{s}}}^{\text{c}}(t) \right) + \\ & \sum_{o^{\text{d}} \in \mathcal{O}^{\text{d}}(t)} \boldsymbol{F}_{io^{\text{d}}}^{\text{a}}(t)\end{aligned} \tag{4}$$

$$T_i(t) = T_i^{\text{s}}(t) + T_i^{\text{c}}(t) + T_i^{\text{m}}(t) \tag{5}$$

In Eq. 4, the first term on the right-hand side corresponds to the motive force on the target victim whose intended velocity at time instant t (i.e., the time derivative of $\boldsymbol{r}_i^{\text{v}}(t)$) is $\boldsymbol{v}_i^{\text{v}}(t)$, and $\boldsymbol{v}_i^0$ is the velocity vector field, which steers the target victim towards an exit of the building. The relaxation time parameter τ_i is an indicative of how fast the target victim reaches the intended speed. The second term in Eq. 4 corresponds to the interactions among victim i and other target victims, including social, contact, and attraction forces. The third term corresponds to the interactions between target victim i and static obstacles in the SaR environment, including the social and contact forces. The last term represents other interactions among the victim and the SaR environment, including the attraction forces with respect to the moving or propagating obstacles (e.g., fire-human repulsion). In Eq. 5, the three terms on the right-hand side are the torques corresponding to the social, contact, and motive forces, respectively. For a more detailed description of the force and torque components in Eqs. 4 and 5, see [68]. In our model, the forces and torques exerted by the other agents are instead exerted by each victim area $A_i(k^{\text{s}})$, which location is indeed available for the robot. Since our SaR problem is formulated in discrete time, a discretized version of Eqs. 2, 3, 4 and 5 formulated as a first-order difference equation for the state vector $\boldsymbol{s}_i^{\text{v}}$ can be used, i.e.,

$$\boldsymbol{s}_i^{\text{v}}(k^{\text{s}} + 1) = f^{\text{v}}\left(\boldsymbol{s}_i^{\text{v}}(k^{\text{s}}), \boldsymbol{F}_i^{\text{total}}(k^{\text{s}}) \right) \tag{6}$$

where:

$$\boldsymbol{s}_i^{\text{v}}(k^{\text{s}}) = \left[\boldsymbol{r}_i^{\text{v}}(k^{\text{s}})^\top, \dot{\boldsymbol{r}}_i^{\text{v}}(k^{\text{s}})^\top \right]^\top \tag{7}$$

$$\boldsymbol{F}_i^{\text{total}}(k^{\text{s}}) = \left[\frac{\boldsymbol{F}_i(k^{\text{s}})^\top}{m_i}, \frac{\boldsymbol{\xi}_i(k^{\text{s}})^\top}{m_i} \right]^\top \tag{8}$$

and:

$$f^{\mathrm{v}}(\zeta_1, \zeta_2) = e^{AT^{\mathrm{s}}}\zeta_1 + \zeta_2 B \int_0^{T^{\mathrm{s}}} e^{Az} dz \tag{9}$$

with:

$$A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 \\ \frac{1}{m_i} & \frac{1}{m_i} \end{bmatrix} \tag{10}$$

and for further details, see [73].

Remark *For the ease of computations, after each evolution of $A_i(k^{\mathrm{s}})$ one may consider the convex hull of the resulting area. However, when the degree of non-convexity is large (i.e., the ratio of the surface of the convex hull and the surface of the original area is larger than a threshold), we allow the area to separate into different sub-areas and then consider the convex hulls of these sub-areas. This means that there may be more than one potential area in the SaR environment that embeds that particular victim. As soon as the victim is detected by the robot, all these sub-areas will be removed from the memory and prediction module of the robot. Moreover, whenever the surface of $A_i(k^{\mathrm{s}}) \cap A_j(k^{\mathrm{s}})$ for two different victims i and j is larger than a specific percentage of the surface of the smaller area, the two victims are assumed to approach each other to move on as a team. Therefore, their approximate areas will be merged.*

Remark *In large-scale implementations, to improve the computation time whenever $C^{\mathrm{p}}(k^{\mathrm{s}}) \cap A_i(k^{\mathrm{s}}) \neq \emptyset$, but the victim is not yet detected (see* Fig. 3*), the intersection will be excluded from the approximate area $A_i(k^{\mathrm{s}})$.*

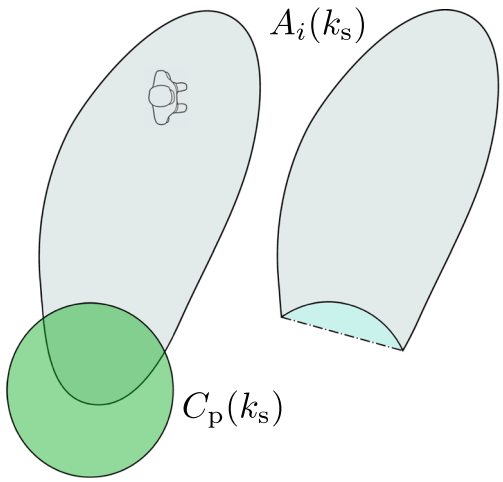

Fig. 3 When the perception field of the SaR robot intersects with the proximate area $A_i(k^{\mathrm{s}})$ of a victim, but does not detect the victim in the intersected area, the intersection will be excluded from the modelled approximate area for the robot to improve the computational efficiency

2.2.3 Formulation of MPC for Coverage-Oriented SaR

The main idea of robust tube-based MPC approach is to decompose the control problem into two problems [74], where first a deterministic nominal MPC problem (considering the nominal models and no disturbances) is solved to determine a desired state sequence and the corresponding control input sequence within the prediction window of the MPC. Next, a feedback-based control problem is formulated that determines a control input increment per control time step in order to keep the realized state sequence as close as possible to the desired one determined by the nominal MPC. For the SaR problem, we formulate the nominal MPC problem (11).

$$\min_{\tilde{\boldsymbol{v}}(k^{\mathrm{c}})} J^{\mathrm{nom}}\left(k^{\mathrm{c}}, \tilde{\boldsymbol{r}}^{\mathrm{rob}}(k^{\mathrm{c}}), \tilde{\boldsymbol{v}}(k^{\mathrm{c}})\right) \tag{11a}$$

s.t. for $\kappa = k^{\mathrm{c}}, \ldots, k^{\mathrm{c}} + N^{\mathrm{p}} - 1$

$$A_i(\kappa + 1) = \mathcal{M}^{\mathrm{Evac}}\left(A_i(\kappa)\right), \text{ for } i = 1, \ldots, N^{\mathrm{v}} \tag{11b}$$

$$\text{if } |C^{\mathrm{p}}(\kappa) \cap A_i(\kappa)| \geq \pi^{\mathrm{c}}|A_i(\kappa)|, \text{ then } A_i(\bar{\kappa}) = \emptyset, \text{ for } \bar{\kappa} = \kappa + 1, \ldots, \kappa + N^{\mathrm{p}} \tag{11c}$$

$$C^{\mathrm{p}}(\kappa + 1) = \mathcal{T}\left(C^{\mathrm{p}}(\kappa), \left[\cos\left(\theta^{\mathrm{rob}}(\kappa)\right), \sin\left(\theta^{\mathrm{rob}}(\kappa)\right)\right]^{\top} v^{\mathrm{rob}}(\kappa)\right) \tag{11d}$$

$$v^{\mathrm{rob}}(\kappa) \leq v^{\mathrm{rob,max}} \tag{11e}$$

$$\Delta v^{\mathrm{rob}}(\kappa) \leq \Delta v^{\mathrm{rob,max}} \tag{11f}$$

$$\Delta\theta^{\mathrm{rob}}(\kappa) \leq \omega^{\mathrm{rob,max}}\tau, \text{ with } \tau \ll T^{\mathrm{c}} \tag{11g}$$

$$x^{\mathrm{rob}}(\kappa) \text{ and } y^{\mathrm{rob}}(\kappa) \text{ evolve according to nominal version of kinematics model by Eq. 1} \tag{11h}$$

$$\boldsymbol{v}(\kappa) = [v^{\mathrm{rob}}(\kappa), \theta^{\mathrm{rob}}(\kappa)]^{\top} \tag{11i}$$

$$\boldsymbol{r}^{\mathrm{rob}\top}(\kappa + 1) \in \mathcal{E} \setminus \left(\mathcal{O}^{\mathrm{s}} \cup \mathcal{O}^{\mathrm{d}}(k)\right) \tag{11j}$$

In Eq. 11, k^{c} is the control time step, N^{p} is the MPC prediction horizon, and tilde is used for a variable to indicate the sequence of that variable across the MPC prediction horizon, e.g., $\tilde{\boldsymbol{r}}^{\mathrm{rob}}(k^{\mathrm{c}})$ is the sequence of the predicted nominal state variables (i.e., coordinates $x^{\mathrm{rob}}(\kappa)$ and $y^{\mathrm{rob}}(\kappa)$) of the robot within the prediction horizon, i.e., for $\kappa \in \{k^{\mathrm{c}}, \ldots, k^{\mathrm{c}} + N^{\mathrm{p}} - 1\}$. Thus, we have $\tilde{\boldsymbol{r}}^{\mathrm{rob}}(k^{\mathrm{c}}) = \left\{\boldsymbol{r}^{\mathrm{rob}}(k^{\mathrm{c}}), \ldots, \boldsymbol{r}^{\mathrm{rob}}(k^{\mathrm{c}} + N^{\mathrm{p}})\right\}$. The optimization variable $\tilde{\boldsymbol{v}}(k^{\mathrm{c}})$ includes the sequence of the nominal control input $\boldsymbol{v}(\kappa)$, which is the vector of the linear velocity $v^{\mathrm{rob}}(\kappa)$ and the angular position $\theta^{\mathrm{rob}}(\kappa)$ of the robot, estimated across the prediction window $\kappa \in \{k^{\mathrm{c}}, \ldots, k^{\mathrm{c}} + N^{\mathrm{p}} - 1\}$, i.e., we have $\tilde{\boldsymbol{v}}(k^{\mathrm{c}}) = \{[v^{\mathrm{rob}}(k^{\mathrm{c}}), \theta^{\mathrm{rob}}(k^{\mathrm{c}})], \ldots, [v^{\mathrm{rob}}(k^{\mathrm{c}} + N^{\mathrm{p}} - 1), \theta^{\mathrm{rob}}(k^{\mathrm{c}} + N^{\mathrm{p}} - 1)]\}$.

Constraint (11b) predicts the evolution and transition of approximate areas A_i according to the mapping $\mathcal{M}^{\text{Evac}}$, which corresponds to the crowd evacuation model explained in Section 2.2.2. Constraint (11c) assesses per control time step whether or not the surface of the intersection area for C^{p} and A_i has reached a specific threshold (given as a percentage of A_i), with $|\cdot|$ giving the surface of a continuous 2-dimensional area and π^{c} a design/tuning parameter; whenever this condition is satisfied, the control system assumes that target victim i has been detected (or more precisely the robot should not search for this victim any more) and thus A_i is considered as a null set afterwards. Constraint (11d) describes the time evolution of the coordinates of the perception field $C^{\text{p}}(k)$ of the robot (i.e., the area that the sensors of the robot can cover, no matter whether or not parts of the area are obstructed by obstacles). Since the shape of the perception field always remains the same (i.e., circular), $\mathcal{T}(\cdot, \boldsymbol{\lambda})$ is a linear mapping that transitions a 2-dimensional area (or a set of coordinates) along the given 2-dimensional vector $\boldsymbol{\lambda}$. In our case, this vector is the linear velocity of the robot. Constraints (11e), (11f) and (11g) account for the physical constraints of the robot, with $\omega^{\text{rob,max}}$ the maximum angular velocity of the robot, $\Delta v^{\text{rob}}(k^{\text{c}}) = v^{\text{rob}}(k^{\text{c}}) - v^{\text{rob}}(k^{\text{c}} - 1)$, and T^{c} the control sampling time (which, in general, is different from the simulation sampling time T^{s}), and τ a fixed constant. For the nominal controller, these constraints are tightened compared to those of the ancillary controller (see [74] for further information). Constraint (11h) describes the evolution of the states of the robot according to the nominal version of the model given by Eq. 1, i.e., when $\delta_x(\kappa)$ and $\delta_y(\kappa)$ are excluded. Constraint (11i) formulates that the vector of nominal control input, i.e., the optimization variables of the nominal MPC, includes the linear velocity and the angular position of the robot. Finally, constraint (11j) allows the robot to move in the free and safe part of the SaR environment (i.e., avoids collision with obstacles). Note that $\mathcal{O}^{\text{s}}$ and $\mathcal{O}^{\text{d}}(k)$ include safety margins between the robot and obstacles and victims. If needed, constraint (11j) may be softened (but never relaxed) by adjusting these safety margins.

The problem is guaranteed to be recursively feasible as explained next. There are hard constraints on the states of the robot via (11j) and on the control inputs via (11f) and (11g). However, the initial configuration, based on Assumption A3, provides initial feasibility for the control problem, which implies recursive feasibility due to the following reasons: According to (11b), targets continuously move according to a physics-based state-space model. Thus, even if their position temporarily obstructs the exit of the robot, the exit will become accessible again after they move. Moreover, upper bound limits on the control inputs (i.e., speed of the robot) only increase the time of reaching the exit, but do not make the problem infeasible.

The objective function in Eq. 11a is given by Eq. 12, that is composed by Eqs. 13 and 14.

$$\begin{aligned} &J^{\text{nom}}\Big(k^{\text{c}}, \tilde{\boldsymbol{r}}^{\text{rob}}(k^{\text{c}}), \tilde{\boldsymbol{v}}(k^{\text{c}})\Big) \\ &= J^{\text{nom,target}}\Big(k^{\text{c}}, \tilde{\boldsymbol{r}}^{\text{rob}}(k^{\text{c}}), \tilde{\boldsymbol{v}}(k^{\text{c}})\Big) \\ &+ J^{\text{nom,coverage}}\Big(k^{\text{c}}, \tilde{\boldsymbol{r}}^{\text{rob}}(k^{\text{c}}), \tilde{\boldsymbol{v}}(k^{\text{c}})\Big) \end{aligned} \tag{12}$$

$$J^{\text{nom,target}}\Big(k^{\text{c}}, \tilde{\boldsymbol{r}}^{\text{rob}}(k^{\text{c}}), \tilde{\boldsymbol{v}}(k^{\text{c}})\Big)$$

$$= \sum_{\kappa=k^{\text{c}}}^{k^{\text{c}}+N^{\text{p}}-1} \Bigg(w_0 \Big(v^{\text{rob}}(\kappa)\Big)^{-1} + \tag{13a}$$

$$w_1 \big|C^{\text{p}}(\kappa) \cap A_1(\kappa)\big|^{-1} + \ldots + w_{N^{\text{v}}} \big|C^{\text{p}}(\kappa) \cap A_{N^{\text{v}}}(\kappa)\big|^{-1} + \tag{13b}$$

$$w_{N^{\text{v}}+1} \big|C^{\text{p}}(\kappa) \cap (A_1(\kappa) \cup \ldots \cup A_{N^{\text{v}}}(\kappa))\big|^{-1} \Bigg) + \tag{13c}$$

$$w_{N^{\text{v}}+2} \left\| \left[x^{\text{rob}}(k^{\text{c}} + N^{\text{p}}), \;\; y^{\text{rob}}(k^{\text{c}} + N^{\text{p}})\right]^{\top} - \left[x^{\text{exit}}, y^{\text{exit}}\right]^{\top} \right\| \tag{13d}$$

$$J^{\text{nom,coverage}}\Big(k^{\text{c}}, \tilde{\boldsymbol{r}}^{\text{rob}}(k^{\text{c}}), \tilde{\boldsymbol{v}}(k^{\text{c}})\Big)$$

$$= \sum_{\kappa=k^{\text{c}}+1}^{k^{\text{c}}+N^{\text{p}}} \sum_{n=k^{\text{c}}}^{\kappa-1} w_{N^{\text{v}}+3} |C^{\text{p}}(\kappa) \cap C^{\text{p}}(n)| \tag{14}$$

The objective function is composed of a stage cost and a terminal cost, where w_j for $j = 0, \ldots, N^{\text{v}}+3$ are weighting factors. In Eq. 12 the nominal economic cost has been formulated, which includes a target-oriented SaR term $J^{\text{nom,target}}(\cdot)$ and a coverage-oriented SaR term $J^{\text{nom,coverage}}(\cdot)$. The target-oriented objective function $J^{\text{nom,target}}(\cdot)$ is expanded in Eq. 13, where (13a)-(13c) represent the target-oriented stage costs, and (13d) corresponds to the target-oriented terminal cost. Moreover, Eq. 14 expands the formulation of the coverage-oriented objective function. Since coverage is a stage property, this objective function is only composed of stage terms. In (13), the first term (13a) reduces the travel time of the robot from its initial position at control time step k^{c} to its target position at the end of the prediction horizon, i.e., at control time step $k^{\text{c}} + N^{\text{p}} - 1$. With the second term (13b), the stage cost maximizes the intersection of the perception field $C^{\text{p}}(\kappa)$ of the robot and every area $A_j(\kappa)$ where the victims are located, for $j = 1, \ldots, N^{\text{v}}$. The third term of the stage cost (13c) maximizes the intersection between $C^{\text{p}}(\kappa)$ and the union of the areas $A_1(\kappa), \ldots, A_{N^{\text{v}}}(\kappa)$. Finally, the terminal cost (13d) - which should become highlighted by adapting $w_{N^{\text{v}}+2}$ when the robot is close to the end of its battery life - assures that the distance between the robot and the exit is terminally minimized. The term in Eq. 14 minimizes the area of the intersection between $C^{\text{p}}(\kappa + i)$ and

$C^{\mathrm{p}}(\kappa), \ldots, C^{\mathrm{p}}(\kappa+i-1)$, for $i=1,\ldots,N^{\mathrm{p}}$, and is used to increase the coverage of the area.

The optimization problem given by Eqs. 11-14 is in general nonlinear and non-convex. While linearization of the problem and deploying quadratic optimizers may be considered, we instead use a well-established nonlinear tube-based approach [74], due to the following reasons: (1) Avoid introducing further model mismatches that may impact the recursive feasibility. (2) Prevent an increase in the computation time, which occurs when the linearization is not efficient. (3) Avoid significant decrease in the precision by oversimplifying the model, where imprecision negatively impacts both the performance and the recursive feasibility. Note that based on Assumption A3 the MPC problem (11)-(14) is feasible by nature.

2.2.4 Dealing with Uncertainties Via Robust Tube-Based MPC

In order to deal with uncertainties in the environment, we implement a robust tube-based version [75] of the nominal MPC proposed in Section 2.2.3. Consider the following nonlinear system subject to bounded additive disturbances $\boldsymbol{w}(k) \in \mathbb{W}$, e.g., the displacement of the robot position considered as non-smoothness of the floor (compare with (1) where $[\delta_x(k^{\mathrm{s}}), \delta_y(k^{\mathrm{s}})]^\top$ is the external disturbance vector):

$$\boldsymbol{\chi}(k+1) = f(\boldsymbol{\chi}(k), \boldsymbol{u}(k)) + \boldsymbol{w}(k) \tag{15}$$

The following ancillary controller will be formulated to obtain a feedback control law in the nonlinear case [74], thus this is the objective function:

$$J^{\mathrm{anc}^*}(\tilde{\boldsymbol{\chi}}(k), \tilde{\boldsymbol{z}}(k)) = \min_{\tilde{\boldsymbol{u}}(k)} J^{\mathrm{anc}}(\tilde{\boldsymbol{\chi}}(k) - \tilde{\boldsymbol{z}}(k), \tilde{\boldsymbol{u}}(k) - \tilde{\boldsymbol{v}}(k)) \tag{16}$$

where the constraints are as in Eq. 11, and this is the control input:

$$\tilde{\boldsymbol{u}}^*(k) \in \arg\min_{\tilde{\boldsymbol{u}}(k)} J^{\mathrm{anc}}(\tilde{\boldsymbol{\chi}}(k) - \tilde{\boldsymbol{z}}(k), \tilde{\boldsymbol{u}}(k) - \tilde{\boldsymbol{v}}(k)) \tag{17}$$

where the symbol $\tilde{\cdot}$ is used with a variable to show the sequence of that variable within the prediction horizon and $\boldsymbol{z}(k)$ and $\boldsymbol{v}(k)$ correspond to the nominal system, i.e.:

$$\boldsymbol{z}(k+1) = f(\boldsymbol{z}(k), \boldsymbol{v}(k)) \tag{18}$$

The ancillary control input will be added to the nominal input $\boldsymbol{v}(k)$ to control system (15). In our case, the disturbances correspond to $\delta_x(k^{\mathrm{s}})$ and $\delta_y(k^{\mathrm{s}})$ in the model for the motion of the robot, i.e., in (1). These deviations may occur due to, e.g., the non-smooth ground on which the robot moves.

3 Case Study

Next we explain the case study, we present the results, and discuss them.

3.1 Setup

The models and the algorithms in this case study have been implemented in MATLAB (R2021a version), ROS (Melodic version) and Gazebo (9.0 version). The victims movement model has been implemented and simulated in, respectively, ROS and Gazebo. The robot simulated for this case study is a TurtleBot 3 Burger [76], which has open source ROS and Gazebo models [77]. The robust tube-based MPC controller has been implemented in MATLAB, which was connected to the robot model in Gazebo in order to simulate it. The PC, on which the experiments have been simulated, had Intel Core i7 processor with 4 cores at 1.80 GHz-2.30 GHz and 16 GB RAM.

The environment of the robot is a partially collapsed building with obstacles (see Section 2.1 for details) that has been simulated in Gazebo. In this scenario, we have a 12 m by 12 m square room, where there are three victims and five static obstacles. While in the formulation we assume obstacles with any shape, in this case study we consider them to be cylinders, with cross sections of radius 0.5 m and their centers positioned at (0.49, 0), (-5, -1), (0, 5), (5, 1) and (2, -5) (see Fig. 4). We suppose that the victims start at positions (-1, -3), (-2, 2) and (4, -2), while the robot only knows an estimate initial area that each victim is positioned in, where these areas are circles of radius 2 m centered around the real positions of the victims. We consider a uniform distribution for the uncertainties about the positions of the victims. This choice is based on [68], which introduces the evacuation model we have adopted for the MPC formulation in this paper. This choice may differ for other implementations, by sampling the initial area $A_i(0)$ of the victims according to another distribution (e.g., Gaussian). The ratio between the area of the environment and the perception field of the robot is 6, therefore the area coverage is substantially larger than the perception range of the robot. The robot starts its mission at coordinates $(-5, -5)$ and ends it at the exit point with coordinates (5, 5). The bounded uncertainties corresponding to the mismatch of the kinematics model of the robot and the reality are selected to satisfy $(\delta_x, \delta_y) \in [-0.1\ \mathrm{m}, 0.1\ \mathrm{m}] \times [-0.1\ \mathrm{m}, 0.1\ \mathrm{m}]$ in a grid

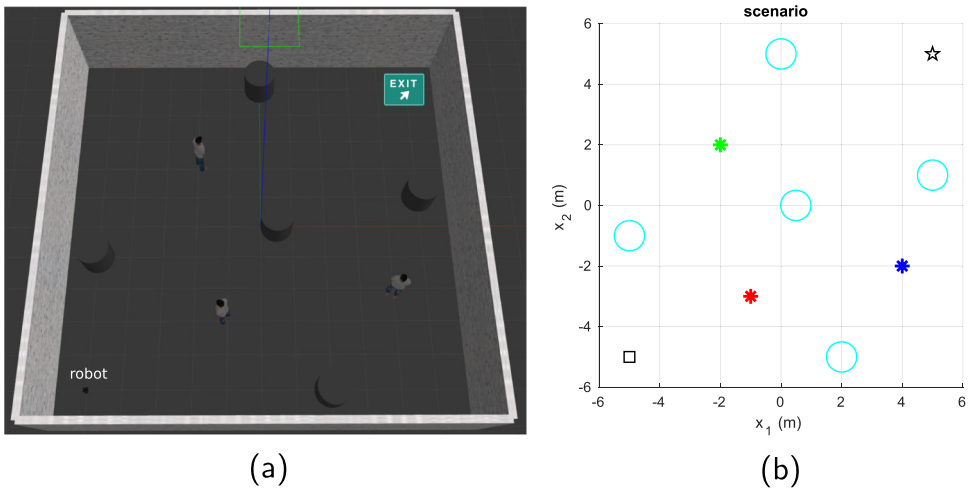

Fig. 4 Schematic view of the simulations in (a) Gazebo and in (b) MATLAB with the three victims (shown via red, green, and blue asterisks), the cylindrical static obstacle (shown via cyan circles), the start point of the robot (shown via a black square) and the exit point (shown via a black star)

of 12 by 12 cells. We consider the values of the uncertainties to remain constant, i.e., every cell corresponds to a fixed uncertainty in the given range. The resulting grid may be interpreted as the non-smoothness map of the environment (the tables are publicly available in the 4TU. ResearchData repository [78]).

The optimization problem given by Eqs. 11-14 for the robot controller has been solved in MATLAB using `fmincon` solver with `sqp` algorithm. The constrained optimization problem of MPC in our (or various other) SaR scenarios is in general non-convex. For non-convex optimization problems, the risk of falling in local optima exists, especially when a gradient-based algorithm is used to solve the optimization problem. Moreover, while in theory global optimization methods (e.g., genetic algorithm) find the global optima, in practice due to the limited time/iterations allocated to solving an optimization problem, such methods may also fail to find global optima. To address this issue, multi-start methods may be used, where several candidate starting values are considered for the optimization variables and the problem is solved for all these starting values. This way, by a proper choice of the starting values, the potential set of solutions is more extensively explored.

In our case studies, we have used a gradient-based approach to solve the optimization problem, because in general gradient-based solvers are faster than most other methods. Multiple starting points were considered per optimization problem, and the realized values of the objective function after solving the optimization problem for all these starting points were compared. The solution that corresponded to the least realized objective value (supposing that a minimization problem is solved) was considered as the solution of MPC. This approach has been used extensively in literature (see, e.g., [79, 80]).

The prediction horizon N^{p} of the MPC controller is 10, with a sampling time of $T^{\mathrm{s}} = 0.2$ s. We assume that the simulation time step k^{s} and the control time step k^{c} are synchronized, thus we use time step k for both. The maximum linear and angular velocities are, respectively, $v^{\mathrm{rob,max}} = 0.7$ m/s and $\omega^{\mathrm{rob,max}} = 2.5$ rad/s. Moreover, we have $\Delta v^{\mathrm{rob,max}} = v^{\mathrm{rob,max}}/2$ and $\Delta\theta^{\mathrm{rob}} = \pi/2 + 0.1$ rad. In our case studies, the weights (which are considered to remain constant during the entire SaR mission) have been selected at the beginning using a trial-and-error approach, such that the selected values are in a meaningful range considering the SaR scenarios that will be simulated, i.e., the candidate initial positions of the victims, the estimated location of the obstacles, and the coordinates of the exit point of the robot.

The trade-off parameter π^{c} for constraint (11c) was tuned via trial-and-error to 0.6: On the one hand, for smaller values of this parameter, it was observed that for various cases the robot over-trusted, based on the estimations of its prediction model, that the target victim will be detected, whereas in reality the robot was missing the victim. This was due to the limited exploration in the potential areas of the target victims by the robot due to a too small value for π^{c}. For larger values for parameter π^{c}, on the other hand, the robot, based on the estimations of its prediction model, would non-efficiently spend extra time exploring the potential areas of the target victims, whereas in reality the target victims were already detected.

The weights in Eq. 14 are put to zero first, because we will consider the influence of the additional term in the objective function for the area coverage separately in Section 3.4.2. The values of the other parameters used in the simulation for forces and torques of the model for the movement of the victims are taken from [68]. Moreover, in the case study we

consider the following additional terminal cost in Eq. 12:

$$
\begin{aligned}
& w_{N^{\mathrm{v}}+4} \left\| \left[x^{\mathrm{rob}}(k^{\mathrm{c}}+N^{\mathrm{p}}), y^{\mathrm{rob}}(k^{\mathrm{c}}+N^{\mathrm{p}}) \right]^{\top} - \left[x^{\mathrm{cg}_1}, y^{\mathrm{cg}_1} \right]^{\top} \right\| + \\
& \quad \ldots + \\
& w_{2N^{\mathrm{v}}+4} \left\| \left[x^{\mathrm{rob}}(k^{\mathrm{c}}+N^{\mathrm{p}}), y^{\mathrm{rob}}(k^{\mathrm{c}}+N^{\mathrm{p}}) \right]^{\top} - \left[x^{\mathrm{cg}_{N^{\mathrm{v}}}}, y^{\mathrm{cg}_{N^{\mathrm{v}}}} \right]^{\top} \right\| + \\
& w_{2N^{\mathrm{v}}+5} \left\| \left[x^{\mathrm{rob}}(k^{\mathrm{c}}+N^{\mathrm{p}}), y^{\mathrm{rob}}(k^{\mathrm{c}}+N^{\mathrm{p}}) \right]^{\top} - \left[x^{\mathrm{cg}_{\mathrm{tot}}}, y^{\mathrm{cg}_{\mathrm{tot}}} \right]^{\top} \right\| \qquad (19)
\end{aligned}
$$

This additional term is used to minimize the distance between the robot and the centers of gravity $\left[x^{\mathrm{cg}_1}, y^{\mathrm{cg}_1}\right]^{\top}, \ldots, \left[x^{\mathrm{cg}_{N^{\mathrm{v}}}}, y^{\mathrm{cg}_{N^{\mathrm{v}}}}\right]^{\top}$ of the victim shapes $A_1, \ldots, A_{N^{\mathrm{v}}}$, and the average center of gravity $\left[x^{\mathrm{cg}_{\mathrm{tot}}}, y^{\mathrm{cg}_{\mathrm{tot}}}\right]^{\top}$ of all these centers of gravity. The reason for adding this term is the following: In theory, with an infinite prediction horizon, for every time step all victims will fall within the prediction horizon of the robot, thus the robust tube-based MPC will be able to compute the intersection with all areas $A_1, \ldots, A_{N^{\mathrm{v}}}$ (see the $\left|C^{\mathrm{p}}(\kappa) \cap A_i(\kappa)\right|^{-1}$ term, for $i = 1, \ldots, N^{\mathrm{v}}$, in Eq. 13). In practice, however, an infinite (or large enough) prediction horizon may make the computations intractable. Thus the term given by (19) is used so that we can select a reasonably small value for N^{p} and reduce the computation time significantly.

The weights for various terms of the objective function were also tuned in advance and by trial and error, as it will be explained next. For the target-oriented objective function, the parameters $w_1, \ldots, w_{N^{\mathrm{v}}}$ in Eq. 13b that weigh the intersection of the area corresponding to the perception field of the robot and the area corresponding to each target victim were tuned to 10^5, whereas $w_{N^{\mathrm{v}}+1}$ in Eq. 13c, i.e., the weight of the term that defines the intersection of the area corresponding to the perception field of the robot and the union of all areas of the target victims was tuned to 10^8. With lower ranges for $w_1, \ldots, w_{N^{\mathrm{v}}}$, the robot was not attracted sufficiently by the individual areas corresponding to the target victims and was mainly moving towards the center of the union area without detecting any individual victims. On the contrary, for much larger ranges for $w_1, \ldots, w_{N^{\mathrm{v}}}$, the robot was purely concentrating on individual target victims, losing the global picture of other target victims. This could result in solutions that were globally far from optimum.

In addition, w_0 is put to 10^{-2}, in order to give priority to finding the victims rather than reducing the mission time, that for the case study is already quite low. Next, $w_{N^{\mathrm{v}}+2}$ is put to 10^3, so that the robot has a reduced priority for going to the exit while it is searching for victims, i.e., compared to $w_1, \ldots, w_{N^{\mathrm{v}}}$ and $w_{N^{\mathrm{v}}+1}$, and higher priority after these terms are no longer relevant because the victims are already found. Finally, for the area coverage, the weight $w_{N^{\mathrm{v}}+3}$ is put to 10^3. Even if lower compared to $w_1, \ldots, w_{N^{\mathrm{v}}}$ and $w_{N^{\mathrm{v}}+1}$, this shows a sufficiently good exploration performance when the term is activated.

The weights of the variation of the terminal state of the robot with respect to the individual centers of gravity in Eq. 19, i.e., $w_{N^{\mathrm{v}}+4}, \ldots, w_{2N^{\mathrm{v}}+4}$, are set to 10^7 and the weight of the variation of the terminal state of the robot with respect to the center of gravity of the union of the shapes, i.e., $w_{2N^{\mathrm{v}}+5}$, is set to 10^4. These values showed, within the range of the parameters and dynamics of the case studies, that a balanced trade-off will be achieved by the robot, to target the individual victims, while considering not to get too far from the rest of the victims.

3.2 Comparison

The proposed robust tube-based approach has been compared in simulation with four state-of-the-art methods, from which the first (Farrokhsiar tube-based MPC) and second (A* MPC) are target-oriented and the third (randomized MPC) and the fourth (boustrophedon motion A*) are coverage-oriented.

The first approach that we compare our results with is the one given in [50]. There, a robust tube-based MPC controller is used to determine a path for the robot that leads to a specific target point. This controller, as in the approach that we propose, is composed of a nominal and an ancillary control, and the uncertainty in the kinematics model of the robot is also the same. However, unlike our robust tube-based MPC controller, the controller in [50] does not have the victims movement model in its constraints, thus the objective function includes a term that minimizes the distance of the robot to the initial positions of the target victims. Moreover, there is no coverage-oriented term in the objective function. This approach is representative of robust methods in literature based on MPC for target-oriented path planning.

The second approach is the one in [81], which uses the A* algorithm to determine a shortest path to a specific target point. An MPC controller is then used to track this path. In order to account for the partial information that is available from the environment for the robot, similarly to [51] at every time step we define an intermediate target point on the border of the perception field of the robot that is the closest to the current target (i.e., a victim or the exit) of the robot. Whenever the robot reaches this intermediate target, the A* algorithm determines a new path, and this procedure continues until the target is reached. This approach is representative of A*-

based methods that are commonly found among standard target-oriented path planning.

The third approach is the one in [82], in which a randomized MPC algorithm is used for path planning. In particular, a number of random samples in the 2D space are generated, and then an algorithm similar to a rapidly exploring random trees (RRT) [83] is used to determine a shortest path that extends through these random points. In this algorithm, the targets are the random points and then the exit. If a random point falls within the forbidden areas, it is discarded since we constrain the algorithm from the beginning. The MPC is used in the planner to generate a collision-free control trajectory. Due to the random selection of the points in the environment, the approach increases the coverage of the area with respect to purely target-oriented approaches. This approach is representative of random-search-based methods, often used as baseline for coverage-oriented path planning. In addition, as a random search strategy, this approach is used as the lower bound benchmark in our experiments.

The fourth approach is the method in [84]. This paper uses the boustrophedon motion algorithm, which mimics the back-and-forth motion of an ox when plowing a field [85], to explore an area as in coverage path planning; when the robot is close to obstacles and cannot continue its current motion, the algorithm determines backtracking points in the visited path and the robot returns to them. Finally, an A* algorithm is used to determine a shortest path to the nearest backtracking point with respect to the current position of the robot; there, the robot starts a new boustrophedon motion, and this procedure is repeated until there are no more unvisited backtracking points. This approach is used as a representative of the heuristic methods that are used in standard coverage path planning.

Finally, we also consider an upper bound benchmark, i.e., an ideal scenario where the robot has perfect knowledge of the environment and the positions of the victims. We implement this benchmark by providing our proposed MPC-based approach with the exact locations of the victims as they appear in the Gazebo simulator.

In all the cases given above, MPC has been used to steer the robot to track the reference path. In addition, for all cases the simulation ends when the robot reaches the exit point.

3.3 Results

In this section, we present the results of the simulations. Figure 5 shows the trajectories of the robot and the 3 victims in the environment, for a case study. In these figures, the trajectory of the robot is shown in black and for the 3 victims the trajectories are shown in red, green, and blue. The following symbols with their corresponding meanings have been used in these figures: Small square, showing the starting positions of the victims and the robot; Star, showing the final position of the victims and the exit point for the robot; Asterisk, showing the position of a victim at the time of being detected by the robot. Whenever the robot detects a victim, the perception field of the robot has also been illustrated. The corresponding videos are publicly available in the 4TU.ResearchData repository [78].

Figure 6 shows the number of victims detected by the robot in time for all the control approaches. Figure 7 presents the percentage of the area that is covered by the perception field of the robot with respect to time during the entire simulation. Figure 8 illustrates via a heat map the number of time steps for which the robot visits the same point of the environment.

Finally, in Table 2 we show the computational cost of the algorithm iterations for all the six approaches of the comparison.

3.4 Discussion

Next, in Section 3.4.1 we discuss and compare the outcome of the different control approaches based on the results given in Section 3.3. After that, in Section 3.4.2 we assess and discuss the performance of the proposed robust tube-based MPC controller, when the objective function includes the area coverage term given in Eq. 14, for a special scenario that properly showcases the importance of the area coverage in SaR missions. Moreover, in Section 3.4.3 we describe and discuss a simulation realized in a special scenario that highlights the non-convexity of the space, in order to show that our approach can be also applied to such non-convex scenarios, despite the risk for the MPC optimization to fall in local minima. Finally, in Section 3.4.4 we present and discuss the results of real-life experiments performed using real robots, comparing our approach to [81] and [82], and we include a discussion on the performance of our tube-based MPC approach compared to two decoupled path planning and trajectory tracking controllers.

3.4.1 Main Scenario

From Fig. 5, our proposed robust tube-based MPC approach guarantees a safe margin in avoiding the obstacles, whereas other control methods (except for [50] which also uses a robust MPC method) result in trajectories that are prone to the risk of collision with the obstacles. In fact, although avoiding the obstacles has been formulated as a constraint for all MPC controllers, due to the absence of a systematic robustness (which is provided by the robust tube-based approach) the robot may crash into an obstacle because of the non-smoothness uncertainty (this occurs, e.g., in Fig. 5 (c) and (d)).

Based on Fig. 5, we observe that there is a trade-off between the time to complete the mission and the number

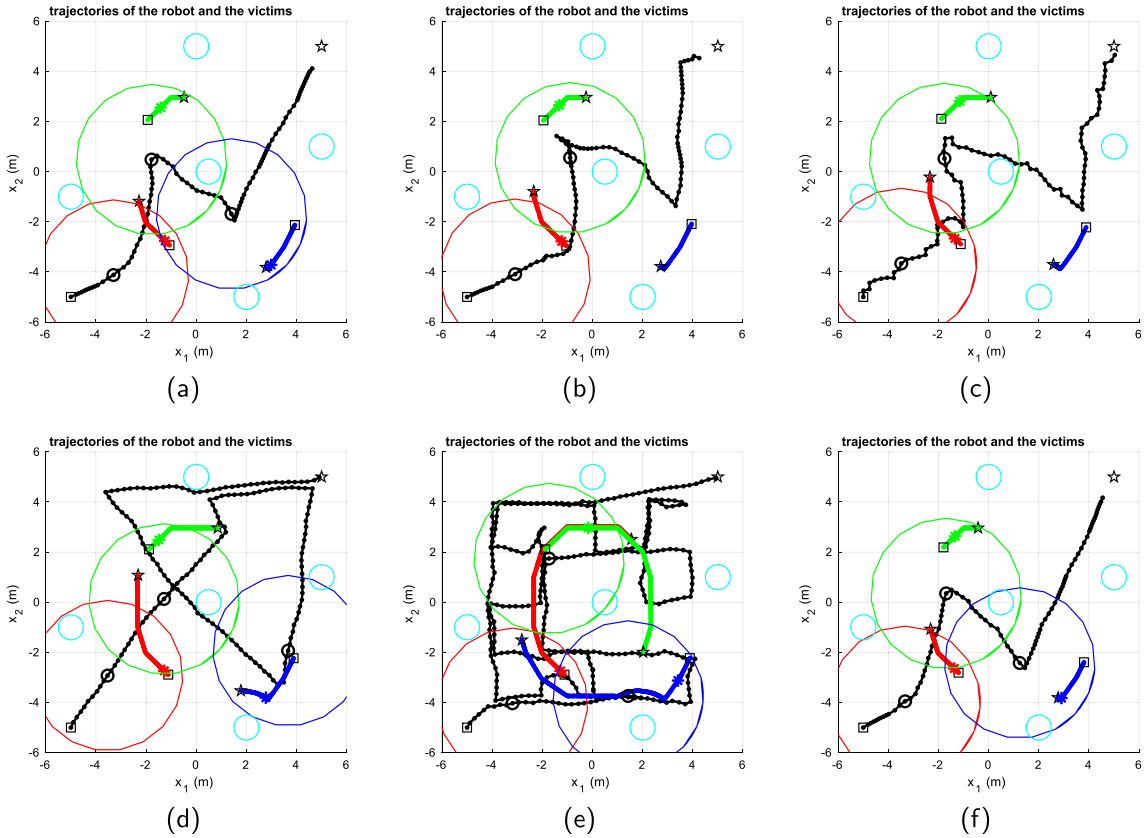

Fig. 5 Plot of the trajectories of the robot (in black) and the three victims (in red, green, and blue) in the environment, with the obstacles shown by small circles (in cyan): Whenever the robot detects a victim, this is illustrated with an asterisk on the trajectory of that victim, as well as by showing the circular perception field of the robot (in the color of the corresponding victim) at the time of detecting the victim. The black squares indicate the starting positions of the victims and the robot, while the black stars illustrate the final position of the victims and the exit point for the robot. Algorithms: (a) Our approach, (b) [50], (c) [81], (d) [82], (e) [84], (f) our approach with perfect information

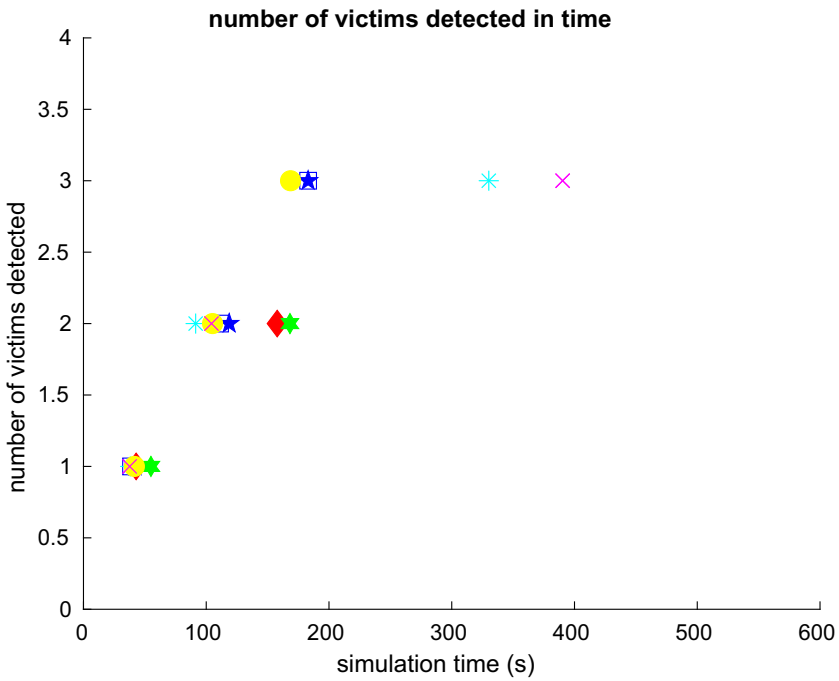

Fig. 6 The number of victims found by the robot: our approach (blue five-pointed star), [50] (red diamond), [81] (green six-pointed star), [82] (cyan asterisk), [84] (magenta cross), and our approach with perfect information (yellow circle). These markers indicate that the victim is inside the perception field (in Gazebo, the robot has detected the real location of the victim); for our approach, the blue square indicates that the victim is detected by the robot (the controller in MATLAB, through the intersection of the areas)

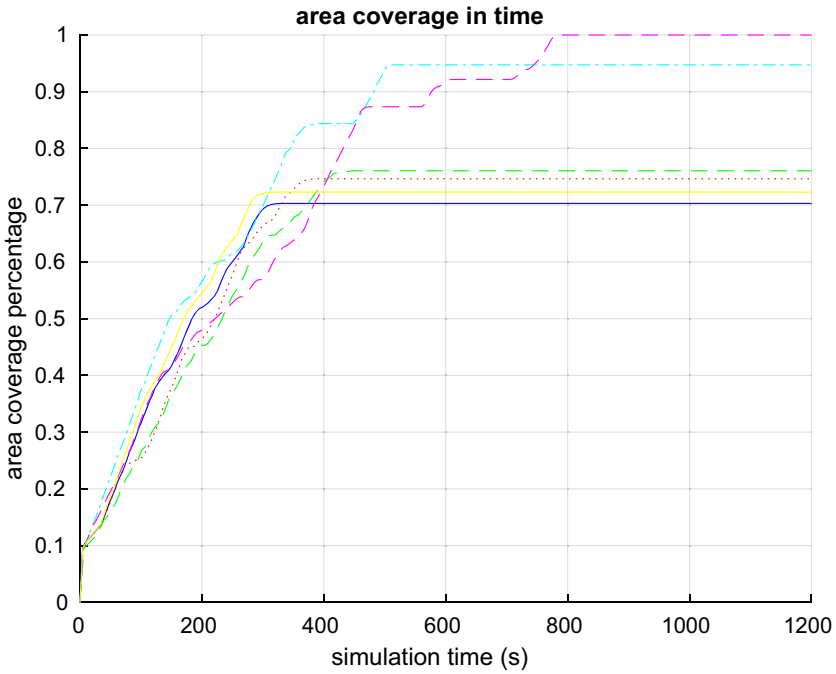

Fig. 7 The percentage of environment area covered in time by our approach (blue solid line), [50] (red dotted line), [81] (green dashed line), [82] (cyan dash-dotted line), [84] (magenta dashed line), and our approach with perfect information (yellow solid line)

of the victims detected. In fact, the robot is able to find more victims with coverage-oriented approaches (see cases (d) and (e)), compared to target-oriented ones (see cases (b) and (c)). This is because the robot normally explores a larger area with coverage-oriented methods in a given time, while with target-oriented methods the mission is accomplished more quickly, after the specific targets are approached. In addition, in the coverage-oriented approaches the trajectories of the robot are more complex, because the robot should avoid the obstacles and the moving victims more times during the mission. Our proposed robust tube-based MPC method (see case (a) in Fig. 5) performs as well as the coverage-oriented

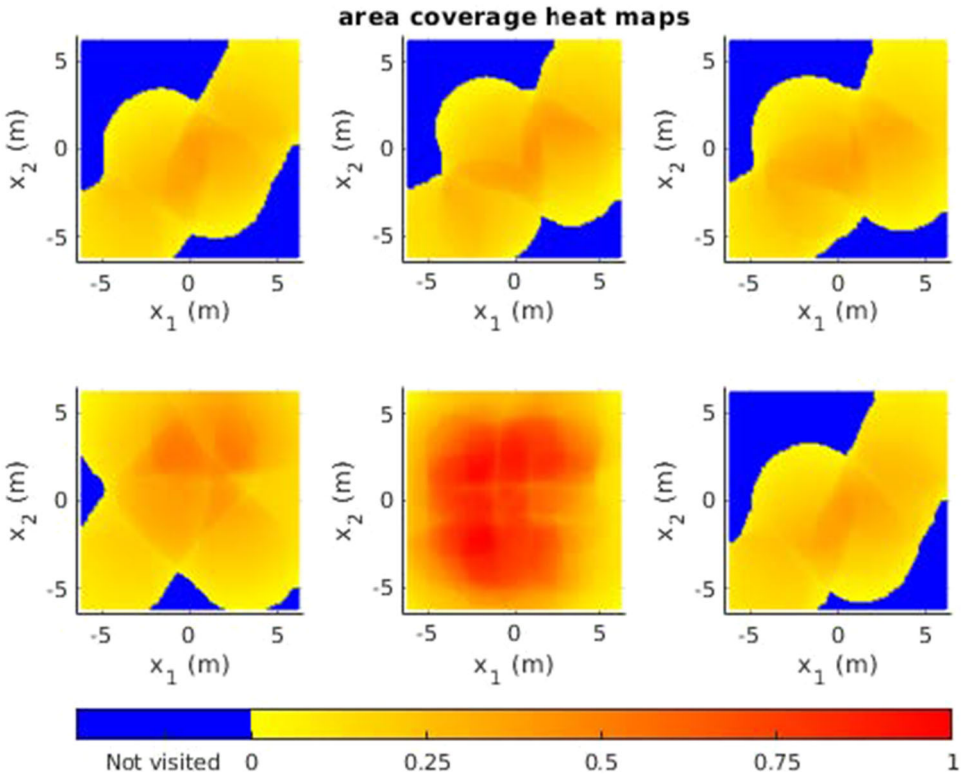

Fig. 8 Heat maps illustrating the intensity of the area coverage, i.e., whenever the robot visits a point for more time steps, the intensity of the heat map varies from 0 to its maximum 1. From top left to bottom right: Our approach, [50, 81, 82, 84], and our approach with perfect information

Table 2 Computational cost in seconds, for the average algorithm iteration, of the approaches of the comparison: (1) Our tube-based MPC, (2) Farrokhsiar tube-based MPC [50], (3) A* MPC [81], (4) Randomized MPC [82], (5) Boustrophedon motion A* [84], and (6) our approach with perfect information

Approach	1	2	3	4	5	6
Computational cost	81.70	65.43	131.13	331.51	220.29	58.51

methods in detecting the victims, while resulting in a simpler trajectory and finishing the mission faster. These results are confirmed also via Fig. 6. In general, with the target-oriented approaches (see the red and green symbols) the robot is not able to find and reach all the victims. This is because the robot encounters victims by chance, while moving towards their initial positions (note that in these cases the robot does not track the victims via their movement model). As expected, with the coverage-oriented approaches (see the blue, cyan and magenta symbols), the robot visits a larger percentage of the environment, and therefore it is likely to find more victims than with the target-oriented methods. Moreover, with our proposed method the robot outperforms the target-oriented approaches in victim detection (see the red, green and blue symbols), and all target-oriented and coverage-oriented methods in speed for finding the victims (i.e., using our approach all victims are detected by time 183.0 s). Only the upper-bound benchmark approach (see the yellow symbol in Fig. 6) is faster than our method, since it has perfect knowledge of the locations of the victims and therefore, can reach them more quickly. Furthermore, in our approach the prediction of the robust tube-based MPC about detecting the victims is closely aligned with real detection of the victims (compare the blue square and blue star in Fig. 6). This implies that the trade-off parameter π^c has properly been tuned.

From Fig. 7 we can compare the area coverage performance of different control approaches. At the beginning of the simulation, all the methods perform similarly. As it is expected, the coverage-oriented approaches (see the cyan and magenta curves) outperform the target-oriented approaches (see the red and green curves) in the long run: In fact, the target-oriented approaches reach a maximum of around 80% area coverage, whereas both coverage-oriented methods reach almost 100% coverage. In the first 400 time steps, our method (see the blue curve) is moderately faster in covering the area compared to the target-oriented approaches. The robot immediately moves towards the exits after detecting all the target victims, thus the area coverage does not increase afterwards. By including the additional coverage-oriented term (see Eq. 14) in the objective function, the area coverage will continue to increase also after detecting the target victims. From the two coverage-oriented approaches, the randomized MPC method [82] outperforms the boustrophedon motion A* method [84] regarding the mission time. This is because the boustrophedon motion coverage algorithm requires the robot to visit the area little by little, whereas random points have a higher chance of falling within the unexplored areas of the environment. In fact, the boustrophedon motion A* approach [84] is the only one that reaches 100% area coverage, thanks to its algorithm that is particularly designed for systematic coverage of the area.

From Fig. 8, via coverage-oriented approaches the robot keeps revisiting some areas of the environment (see the two heat maps at the bottom of the figure), while in our approach and in the target-oriented approaches the robot visits the area once (see the three heat maps at the top of the figure). On the one hand, the behaviour of the coverage-oriented approaches can be preferable in presence of dynamic obstacles or dynamic targets. On the other hand, a trade-off between the time efficiency and the additional information gained by revisiting the areas should be considered.

Comparing the various approaches with the performance of the upper bound benchmark, we show that our approach is the closest to it since it is the fastest in covering new areas (see Fig. 7), due to the combination of target and coverage-oriented objectives, and in detecting the victims (see Fig. 6), since we include an accurate prediction model for the movement of the victims.

For the computation times, Table 2 shows that our approach outperforms A* method [81] and both coverage-oriented approaches. Only Farrokhsiar tube-based MPC [50] and the upper-bound benchmark method are faster than our approach, because unlike our MPC controller, they do not consider the evolution of the 2D areas assigned to each victim in the optimization loop. Since our main focus is on using MPC for optimizing competing costs and handling hard constraints, we analyze the MPC in comparison with the other approaches to showcase its performance, and therefore avoiding to aim for faster computations. The reported results are impacted by the usage of laptops with limited resources that run micro-simulations. In real life, instead, more advanced hardware can be used and micro-simulators are not needed. In addition, using fast MPC methods (see, e.g., [86, 87] about fast MPC alternative methods), the computation time could be significantly improved. There already exists vast research that discusses improvement of computational efficiency (see,

e.g., [88, 89]) and analysis of feasibility of MPC (see, e.g., [90, 91]). Such methods may be used for the MPC formulation that is proposed in this paper, in real-life applications to combined coverage and target-oriented SaR, where fast computation and feasibility should be guaranteed.

3.4.2 Special Scenario for Area Coverage Term

In order to assess the effect of adding the additional area coverage term (i.e., putting the weights of Eq. 14 to non-zero) in the objective function (12), we have simulated an additional specific scenario, in which the robot has to visit three static victims before going to the exit (see Fig. 9 where the position of the victims are shown by colored asterisks). In this scenario, we have a 20 m by 20 m square room, where there are three static victims with initial positions $(0, 8.3)$, $(0, 0)$ and $(0, -8.3)$ and no obstacles. The robot starts its mission at coordinates $(-8, 8)$ and ends at the exit point at coordinates $(8, -8)$. This configuration has been selected because after reaching a certain victim, the robot has additional space, especially at the left and at the right of the victims, that it can explore. In fact, with the additional coverage term, the robot continues on its direction after one victim is detected, to explore more area, until the border of the scenario is reached, where the coverage term is deactivated and therefore the robot goes to the following victim. In particular, from Fig. 10 we can show that the additional term gives a gain in coverage percentage of 29.3%. Finally, in Table 3 we show the computational cost of the MPC minimizations for our approach with and without the additional area coverage term: Without the coverage term, the MPC is more then five times faster, and therefore there is a trade-off in selecting computational performance and area coverage amount. Note that the computation times are not comparable with respect to those in Table 2 because we do not employ multi-start in this optimizations.

3.4.3 Special Scenario for Addressing the Non-Convexity

We have designed a scenario, where due to the configuration of the obstacles (as it is shown in Fig. 11), the resulting constrained optimization problem is obviously non-convex. This scenario considers a 6×6 m^2 room, where two victims are initially located at $(-1, 1)$ and $(1.5, -0.5)$, and various obstacles create non-convex shapes. The robot starts its mission from position $(-2.5, -2.5)$ and ends the mission at the exit point located at $(2.5, 2.5)$.

The optimization problem was run for 5 different starting points per iteration, where one starting point is the previous solution, while the other four starting points are selected randomly. As Fig. 11 shows, despite a highly non-convex space, the robot executes its mission by successfully detecting both victims, and by reaching the exit afterwards, while avoiding the obstacle traps. Moreover, Table 4 shows the realized values of the objective function for these 5 optimization starting

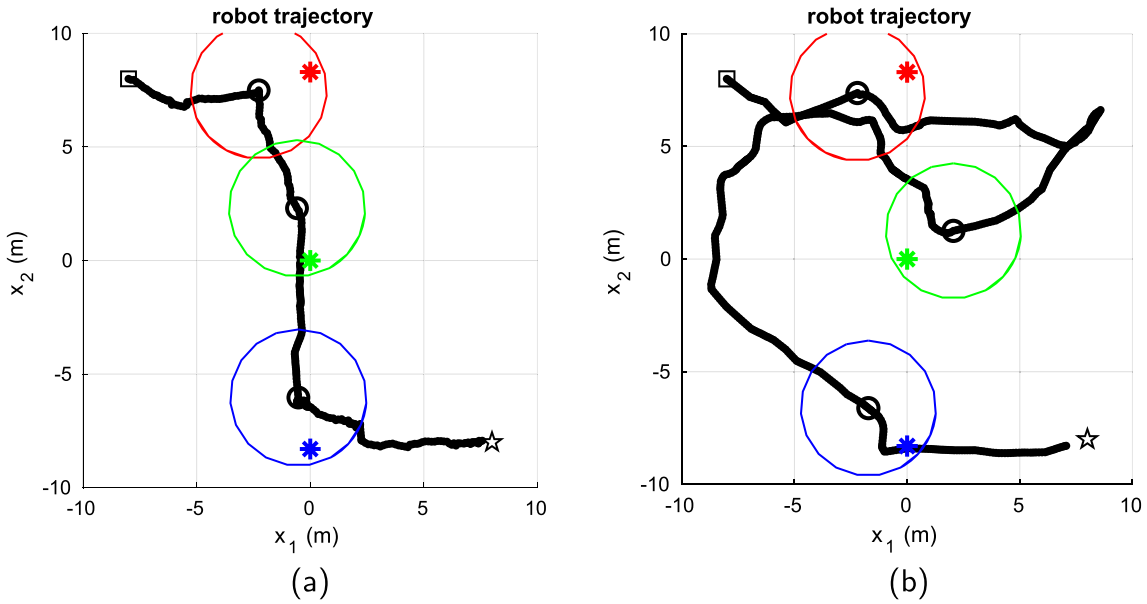

Fig. 9 The robot trajectory (in black) with the position of the three victims (red, green, and blue asterisks); whenever the robot detects a victim, the circular perception field of the robot is illustrated in the color of the detected victim. (a) Proposed robust tube-based MPC approach without the area coverage term. (b) Proposed robust tube-based MPC approach with the area coverage term

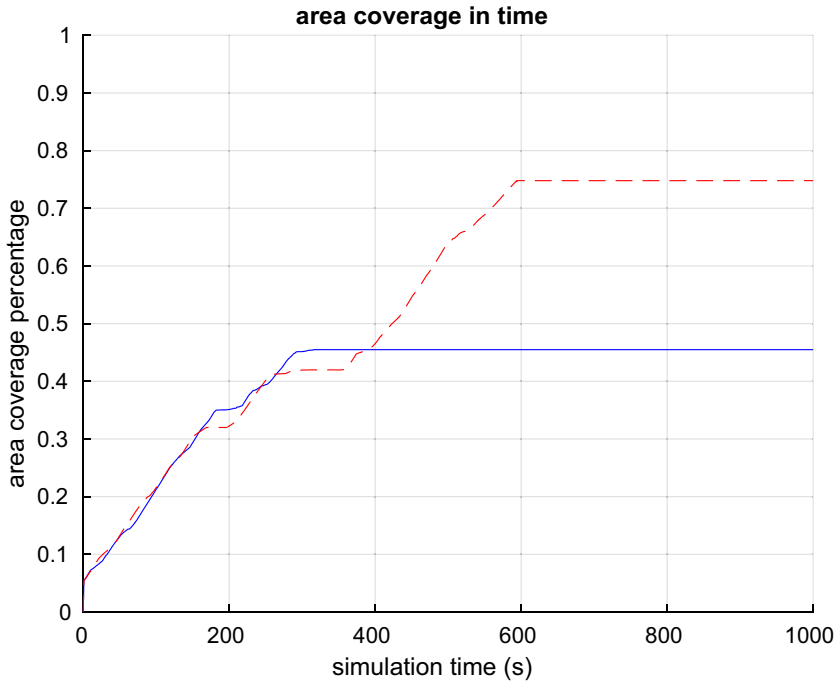

Fig. 10 Percentage of the area covered in the special scenario: proposed robust tube-based MPC approach without the area coverage term (blue solid line) and proposed robust tube-based MPC approach with the area coverage term (red dashed line)

points, for 3 randomly selected optimization iterations. The solution corresponding to the value in bold fonts (i.e., the least of the 5 values) is selected per iteration.

3.4.4 Real-Life Experiments

In order to show the applicability of our approach beyond simulations, especially when real-life uncertainties, disturbances, and measurement noise exists, we designed and performed experiments in real life, using the same software (MATLAB, ROS) and laptop, as described in Section 3.1. We used a TurtleBot 3 Burger [76] to represent a SaR robot, and additional robots, i.e., iRobot Create 3 [92], to move according to the victims motion model and to represent the victims. The movements of the iRobot Create are aligned with the movements of the victims. More specifically, the iRobot Create first receives commands on the rotation angle, and then on the linear velocity, where these values are corresponding to those computed with the victims motion model. The environment and the robots are shown in Fig. 12.

Three scenarios were simulated in a room of size 3.5 × 5.2 m^2, with three obstacles at locations (0, 0), (0.5, 2) and (-0.5, -2), and a victim with a different initial position per scenario, i.e., (1.5, -0.5) for scenario 1, (-1, 1) for scenario 2, and (1.25, 0.75) for scenario 3, with a circular uncertain positioning area of radius 0.5 m. The robot started its mission at coordinates $(-1.25, -2.1)$ and ended the mission at the exit with coordinates $(1.25, 2.1)$. Moreover, the perception field of the robot had a radius of 1.5 m.

With regard to the source of uncertainties in the real-life experiments, the drift of the odometry measurements based on the motion of the wheels generates disturbances in computing the position of the robot. In addition, another source of uncertainties is due to the misalignment of the map of the environment generated by the robot and the real configuration of the scenario. These uncertainties also affect the localization of the robot. Finally, there may be mismatches between the model of the TurtleBot that is used in ROS for sending the motion commands and the actual behavior of the robot.

We have presented the results of the experiments in Tables 5, 6 and 7 (for scenarios 1, 2 and 3, respectively), where the time of detecting the victim and the average computation time per iteration are given. The results are presented and compared for our approach, and for the target-oriented and coverage-oriented approaches given in, respectively, [81] and [82]. For all the three scenarios, our approach detects the victim (up to 50.0% with respect to the slowest method)

Table 3 Computational cost in seconds, for the average MPC minimization, of our proposed approach with and without the additional coverage term

Our approach without coverage term	Our approach with coverage term
6.01	32.21

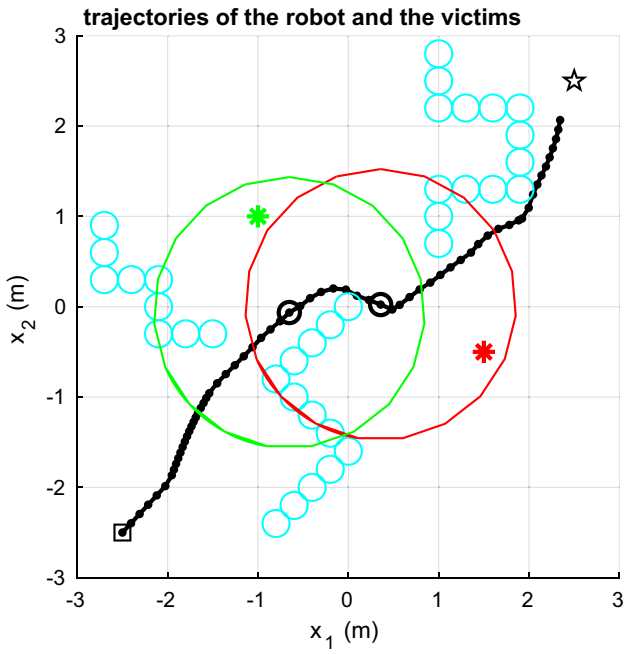

Fig. 11 A highly non-convex SaR scenario: The robot trajectory is shown in black and the real positions of the two victims that are determined via the micro-simulator are shown with a red and a green asterisk. Whenever the robot detects a victim, the circular perception field of the robot has been illustrated in the same color used for illustrating the position of the detected victim

Fig. 12 A view of the scenario of the experiments, with the robots (the TurtleBot used as the SaR robot is in the middle and the iCreate3 robot used as the victim is at the side) and the obstacles (white buckets)

faster. In all the 3 scenarios our approach succeeds in detecting the victim, whereas each of the other 2 approaches fails once. This is due to the fact that our approach has a victim movement model and incorporates the uncertainties in the trajectories of these victims, which together raise the chances of detecting the victims, in a more time-efficient way. About the average computation time, our approach is up to 76.1% faster than the two other approaches. This is in accordance with Table 2 for the simulations results. Corresponding videos of the experiments are publicly available via the 4TU.ResearchData repository [78].

We have also performed an experiment to compare the area coverage percentage of the three approaches, when the coverage term (i.e., Eq. 14) has been activated for our approach. In this case, a static victim is positioned at $(-1, 0)$, and the obstacles are removed. The starting and exit coordinates of the robot are, respectively, $(-1.25, -2.1)$ and $(1.25, -2.1)$. The results for scenario 4 are shown in Table 8, where our approach covers, respectively, 24.7% and 13.6% more than the approaches in [81] and [82].

We have analyzed the computation times of the approach using both nominal and ancillary controllers (therefore, the complete tube-based MPC approach), and only nominal controller (therefore decoupled path planning and trajectory tracking approach). For each of the three scenarios, considering the mean computation time of all the MPC optimization iterations, we have:

- Scenario 1: 2.5086 s (both controllers) and 2.1926 s (only nominal controller).
- Scenario 2: 1.4663 s (both controllers) and 1.3298 s (only nominal controller).
- Scenario 3: 2.6402 s (both controllers) and 2.7304 s (only nominal controller).

Therefore, our combined approach is only 12, 60% slower than the decoupled approach in the first scenario and only 9, 31% slower in the second scenario, while it is 3.30% faster in the third. This is due to the fact that the outputs of the optimizations are different in the two cases, so a trajectory may be longer in a scenario with a certain controller, and shorter in another one. Hence, there can be a benefit of using a combined approach rather than a decoupled approach, depending on the specific scenario.

Table 4 Values of the objective function of the MPC in the nonconvex scenario for three indicated iterations, and the points that are selected as minima are shown in bold

Iteration number	10	20	35
Cost function values	**0.0013**, 0.77, 0.24, 1.15, 0.29	**-0.0037**, 0.98, 0.36, 1.15, 0.32	**-0.0066**, 0.28, 0.097, 0.58, 0.28

Table 5 Time step when the victim is detected, as well as the average computation time per optimization iteration, for scenario 1: The results are shown for our approach, for target-oriented A* MPC [81] and for coverage-oriented randomized MPC [82]

	Our tube-based MPC	A* MPC [81]	Randomized MPC [82]
Time step of victim detection	10	12	–
Computational cost	12.9 s	44.0 s	45.7 s

Table 6 Time step when the victim is detected, as well as the average computation time per optimization iteration, for scenario 2: The results are shown for our approach, for target-oriented A* MPC [81] and for coverage-oriented randomized MPC [82]

	Our tube-based MPC	A* MPC [81]	Randomized MPC [82]
Time step of victim detection	15	–	17
Computational cost	19.9 s	53.8 s	37.4 s

Table 7 Time step when the victim is detected, as well as the average computation time per optimization iteration, for scenario 3: The results are shown for our approach, for target-oriented A* MPC [81] and for coverage-oriented randomized MPC [82]

	Our tube-based MPC	A* MPC [81]	Randomized MPC [82]
Time step of victim detection	13	17	26
Computational cost	14.5 s	60.6 s	30.0 s

Table 8 Area coverage percentage for scenario 4: The results are shown for our approach, A* MPC [81] and Randomized MPC [82]

	Our tube-based MPC	A* MPC [81]	Randomized MPC [82]
Area coverage percentage	74.6 %	49.9 %	61.0 %

4 Conclusions and Future Topics

This paper presented a combined target-oriented and coverage-oriented path planning approach based on robust tube-based implementation of a novel formulation of MPC for a ground SaR robot in a scenario with dynamic targets. The controller determines an optimal path to the moving victims, optimizing the mission time and the area coverage, while dealing with uncertainties. In our case study, we have shown that our method outperforms two target-oriented and two coverage-oriented state-of-the-art methods in victim detection efficiency, area coverage, and mission completion time, while being robust to uncertainties.

In the future, in order to address more indoor scenarios, we propose to include fire in the simulations and a fire propagation model as prediction model for MPC: In particular, a fire model based on cellular automata will be incorporated and we will study how our approach performs under different conditions of fire propagation based on wind speed and direction, structure and materials of the buildings, and different propagation times from ignition points. Moreover, this work can be expanded to a multi-robot system: We will analyze how the proposed MPC formulation can be used in a distributed or a decentralized architecture, where considering the exchange of information among all the robots, among clusters of them, or no communication; in this case, how the method will scale with the number of robots will be studied and clarified. Furthermore, running simulations in photorealistic environments, for example by using Gazebo environments provided by Amazon Web Services, is recommended. We also propose to extend the case study benchmarks by considering a state-of-the-art learning-based approach to compare against. Another topic for future research is to implement a simplified formulation of the given MPC problem (e.g., linearized) and compare the performance and computations time with when nonlinear MPC is solved directly. Stability analysis is another topic for future work. Assumption A4 (i.e., perfect knowledge of the states of the robot) may be relaxed in the future by integrating of a state estimator within our MPC approach. In addition, in view of the promising results from the computer-based simulations and experiments with real robots, in order to tackle the computation time for more complex SaR cases, fast MPC methods may be adopted in the future for the proposed approaches of this paper. Finally, we can apply one of the techniques that are available in literature in order to deal with the fact that the decision step of the robot takes several seconds while the victims are moving according to their model, so that we can synchronize them.

Author Contributions Author M. Baglioni contributed to defining the SaR scenario and designing and implementing the experiments. Authors M. Baglioni and A. Jamshidnejad contributed to the analysis and interpretation of the results, development of the theoretical contributions, formulation of the proposed approach, and composition of the manuscript. Author M. Baglioni prepared the first draft of the manuscript. Author A. Jamshidnejad supervised the study design, and has reviewed the draft. Author M. Baglioni 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 supported jointly by the TU Delft AI Labs program - as a part of the AI*MAN lab research - and by the NWO Talent Program Veni project "Autonomous drones flocking for search-and-rescue" (18120), which has been financed by the Netherlands Organization for Scientific Research (NWO).

Data Availability We provide two tables with the values of the non-smoothness map, for each coordinate x and y, and videos corresponding to plots in Fig. 5 in the 4TU. Research Data repository [78].

Declarations

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

References

1. Shah, B., Choset, H.: Survey on urban search and rescue robots. Journal of the Robotics Society of Japan. **22**(5), 582–586 (2004)
2. Liu, J., Wang, Y., Li, B., Ma, S.: Current research, key performances and future development of search and rescue robots. Front. Mech. Eng. China **2**(4), 404–416 (2007)
3. Tanzi, T.J., Chandra, M., Isnard, J., Camara, D., Sébastien, O., Harivelo, F.: Towards "drone-borne" disaster management: future application scenarios, in XXIII ISPRS Congress, Commission VIII (Volume III-8), vol. 3 (Copernicus GmbH, 2016), pp. 181–189
4. Davids, A.: Urban search and rescue robots: from tragedy to technology. IEEE Intell. Syst. **17**(2), 81–83 (2002)
5. Murphy, R.R., Tadokoro, S., Nardi, D., Jacoff, A., Fiorini, P., Choset, H., Erkmen, A.M.: Search and Rescue Robotics (Springer Berlin Heidelberg, Berlin, Heidelberg, 2008), pp. 1151–1173. https://doi.org/10.1007/978-3-540-30301-5_51
6. 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)
7. 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)

8. Rawlings, J.B., Mayne, D.Q., Diehl, M.: Model predictive control: theory, computation, and design, vol. 2. Nob Hill Publishing Madison, WI (2017)
9. Bemporad, A., Morari, M.: Robust model predictive control: a survey, in Robustness in Identification and Control, pp. 207–226. Springer, London, U.K. (1999)
10. Scokaert, P.O.M., Mayne, D.Q.: Min-max feedback model predictive control for constrained linear systems. IEEE Trans. Automat. Contr. **43**(8) (1998)
11. Mayne, D.Q., Seron, M.M., Raković, S.V.: Robust model predictive control of constrained linear systems with bounded disturbances. Automatica **41**, 219–224 (2005)
12. Ryan, A., Hedrick, J.K.: A mode-switching path planner for UAV-assisted search and rescue. In: Proceedings of the 44th IEEE Conference on Decision and Control (IEEE, 2005), pp. 1471–1476
13. Berger, J., Lo, N.: An innovative multi-agent search-and-rescue path planning approach. Computers & Operations Research. **53**, 24–31 (2015)
14. Colas, F., Mahesh, S., Pomerleau, F., Liu, M., Siegwart, R.: 3D path planning and execution for search and rescue ground robots. In: 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IEEE, 2013), pp. 722–727
15. Blackmore, L., Ono, M., Bektassov, A., Williams, B.C.: A probabilistic particle-control approximation of chance-constrained stochastic predictive control. IEEE Trans. Rob. **26**(3), 502–517 (2010)
16. Huang, Z., Liu, Q., Liu, J., Huang, B.: A comparative study of model approximation methods applied to economic MPC. The Canadian Journal of Chemical Engineering. **100**(8), 1676–1702 (2022)
17. Hertneck, M., Köhler, J., Trimpe, S., Allgöwer, F.: Learning an approximate model predictive controller with guarantees. IEEE Control Systems Letters. **2**(3), 543–548 (2018)
18. Khadem, M., O'Neill, J., Mitros, Z., Cruz, L.D., Bergeles, C.: Autonomous steering of concentric tube robots via nonlinear model predictive control. IEEE Trans. Rob. **36**(5), 1595–1602 (2020)
19. Skjong, E., Nundal, S.A., Leira, F.S., Johansen, T.A.: Autonomous search and tracking of objects using model predictive control of unmanned aerial vehicle and gimbal: hardware-in-the-loop simulation of payload and avionics. In 2015 International Conference on Unmanned Aircraft Systems (ICUAS) (IEEE, Denver, CO, 2015), pp. 904–913
20. Nattero, C., Recchiuto, C.T., Sgorbissa, A., Wanderlingh, F.: Coverage algorithms for search and rescue with UAV drones. In: Artificial Intelligence, Workshop of the XIII AI* IA Symposium on, vol. 12 (2014)
21. Galceran, E., Carreras, M.: A survey on coverage path planning for robotics. Robot. Auton. Syst. **61**(12), 1258–1276 (2013)
22. Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., Leonard, J.J.: Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. IEEE Trans. Rob. **32**(6), 1309–1332 (2016)
23. Mur-Artal, R., Montiel, J.M.M., Tardos, J.D.: ORB-SLAM: a versatile and accurate monocular SLAM system. IEEE Trans. Rob. **31**(5), 1147–1163 (2015)
24. Chandarana, M., Hughes, D., Lewis, M., Sycara, K., Scherer, S.: Planning and monitoring multi-job type swarm search and service missions. Journal of Intelligent & Robotic Systems. **101**, 1–14 (2021)
25. Kashino, Z., Nejat, G., Benhabib, B.: Aerial wilderness search and rescue with ground support. Journal of Intelligent & Robotic Systems. **99**, 147–163 (2020)
26. Khamis, A.M., Elmogy, A.M., Karray, F.O.: Complex task allocation in mobile surveillance systems. Journal of Intelligent & Robotic Systems. **64**, 33–55 (2011)
27. Husain, Z., Zaabi, A.A., Hildmann, H., Saffre, F., Ruta, D., Isakovic, A.F.: Search and rescue in a maze-like environment with ant and dijkstra algorithms (2021). arXiv:2111.08882
28. Cooper, J.R.: Optimal Multi-Agent Search and Rescue Using Potential Field Theory, in AIAA Scitech 2020 Forum (2020), p. 0879
29. Din, A., Jabeen, M., Zia, K., Khalid, A., Saini, D.K.: Behavior-based swarm robotic search and rescue using fuzzy controller. Computers & Electrical Engineering. **70**, 53–65 (2018)
30. Kumar, G., Anwar, A., Dikshit, A., Poddar, A., Soni, U., Song, W.K.: Obstacle avoidance for a swarm of unmanned aerial vehicles operating on particle swarm optimization: a swarm intelligence approach for search and rescue missions. J. Braz. Soc. Mech. Sci. Eng. **44**(2), 1–18 (2022)
31. Dubé, R., Gawel, A., Cadena, C., Siegwart, R., Freda, L., Gianni, M.: 3D localization, mapping and path planning for search and rescue operations. In: 2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR) (IEEE, 2016), pp. 272–273
32. Dang, T., Mascarich, F., Khattak, S., Papachristos, C., Alexis, K.: Graph-based path planning for autonomous robotic exploration in subterranean environments. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE, 2019), pp. 3105–3112
33. de Koning, C., Jamshidnejad, A.: 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. **107**(140) (2023)
34. 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)
35. Mohseni, F., Doustmohammadi, A., Menhaj, M.B.: Distributed model predictive coverage control for decoupled mobile robots. Robotica **35**(4), 922–941 (2017)
36. Ibrahim, M., Matschek, J., Morabito, B., Findeisen, R.: Hierarchical model predictive control for autonomous vehicle area coverage. IFAC-PapersOnLine. **52**(12), 79–84 (2019)
37. Ibrahim, M., Matschek, J., Morabito, B., Findeisen, R.: Improved area covering in dynamic environments by nonlinear model predictive path following control. IFAC-PapersOnLine. **52**(15), 418–423 (2019)
38. Agarwal, S., Akella, S.: Area coverage with multiple capacity-constrained robots. IEEE Robotics and Automation Letters. **7**(2), 3734–3741 (2022)
39. Guruprasad, K.R., Ranjitha, T.D.: CPC algorithm: exact area coverage by a mobile robot using approximate cellular decomposition. Robotica **39**(7), 1141–1162 (2021)
40. Carr, C., Wang, P.: Fast-spanning ant colony optimisation (FaSACO) for mobile robot coverage path planning (2022). arXiv:2205.15691
41. Juan, V.S., Santos, M., Andújar, J.M.: Intelligent UAV map generation and discrete path planning for search and rescue operations. Complexity. **2018** (2018)
42. 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)
43. Liu, Y., Nejat, G.: Multirobot cooperative learning for semiautonomous control in urban search and rescue applications. Journal of Field Robotics. **33**(4), 512–536 (2016)
44. 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. Journal of Intelligent & Robotic Systems. **94**(3), 669–686 (2019)

45. Apuroop, K.G.S., Le, A.V., Elara, M.R., Sheu, B.J.: Reinforcement learning-based complete area coverage path planning for a modified hTrihex robot. Sensors. **21**(4), 1067 (2021)
46. Saha, O., Ren, G., Heydari, J., Ganapathy, V., Shah, M.: Deep reinforcement learning based online area covering autonomous robot. In: 2021 7th International Conference on Automation, Robotics and Applications (ICARA) (IEEE, Prague, Czech Republic, 2021), pp. 21–25
47. Saha, O., Ren, G., Heydari, J., Ganapathy, V., Shah, M.: Online area covering robot in unknown dynamic environments. In: 2021 7th International Conference on Automation, Robotics and Applications (ICARA) (IEEE, Prague, Czech Republic, 2021), pp. 38–42
48. Carron, A., Zeilinger, M.N.: Model predictive coverage control. IFAC-PapersOnLine. **53**(2), 6107–6112 (2020)
49. 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)
50. 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)
51. Jamshidnejad, A., Frazzoli, E.: Adaptive optimal receding-horizon robot navigation via short-term policy development. In: 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV) (IEEE, Singapore, 2018), pp. 21–28
52. Chung, T.H., Hollinger, G.A., Isler, V.: Search and pursuit-evasion in mobile robotics: a survey. Auton. Robot. **31**, 299–316 (2011)
53. Pierson, A., Rus, D.: Distributed target tracking in cluttered environments with guaranteed collision avoidance. In: 2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS) (IEEE, 2017), pp. 83–89
54. Sani, M., Robu, B., Hably, A.: Pursuit-evasion games based on game-theoretic and model predictive control algorithms. In: 2021 International Conference on Control, Automation and Diagnosis (ICCAD) (IEEE, 2021), pp. 1–6
55. Sani, M., Robu, B., Hably, A.: Limited information model predictive control for pursuit-evasion games. In: 2021 60th IEEE Conference on Decision and Control (CDC) (IEEE, 2021), pp. 265–270
56. Simone, D.D., Scianca, N., Ferrari, P.P., Lanari, L., Oriolo, G.: MPC-based humanoid pursuit-evasion in the presence of obstacles, in 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE, 2017), pp. 5245–5250
57. Chen, T., Gupta, S., Gupta, A.: Learning exploration policies for navigation (2019). arXiv:1903.01959
58. Tan, A.H., Narasimhan, S., Nejat, G.: 4CNet: a confidence-aware, contrastive, conditional, consistency model for robot map prediction in multi-robot environments (2024). arXiv:2402.17904
59. Niroui, F., Zhang, K., Kashino, Z., Nejat, G.: Deep reinforcement learning robot for search and rescue applications: exploration in unknown cluttered environments. IEEE Robotics and Automation Letters. **4**(2), 610–617 (2019)
60. Chaplot, D.S., Gandhi, D., Gupta, S., Gupta, A., Salakhutdinov, R.: Learning to explore using active neural SLAM (2020). arXiv:2004.05155
61. Tan, A.H., Bejarano, F.P., Zhu, Y., Ren, R., Nejat, G.: Deep reinforcement learning for decentralized multi-robot exploration with macro actions. IEEE Robotics and Automation Letters. **8**(1), 272–279 (2022)
62. Devo, A., Mezzetti, G., Costante, G., Fravolini, M.L., Valigi, P.: Towards generalization in target-driven visual navigation by using deep reinforcement learning. IEEE Trans. Rob. **36**(5), 1546–1561 (2020)
63. Mezghan, L., Sukhbaatar, S., Lavril, T., Maksymets, O., Batra, D., Bojanowski, P., Alahari, K.: Memory-augmented reinforcement learning for image-goal navigation. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE, 2022), pp. 3316–3323
64. Fung, A., Wang, L.Y., Zhang, K., Nejat, G., Benhabib, B.: Using deep learning to find victims in unknown cluttered urban search and rescue environments. Current Robotics Reports. **1**, 105–115 (2020)
65. Wang, H., Tan, A.H., Nejat, G.: NavFormer: a transformer architecture for robot target-driven navigation in unknown and dynamic environments. IEEE Robotics and Automation Letters (2024)
66. Mohamed, S.C., Fung, A., Nejat, G.: A multirobot person search system for finding multiple dynamic users in human-centered environments. IEEE Transactions on Cybernetics. **53**(1), 628–640 (2022)
67. Yokoyama, N.H., Ha, S., Batra, D., Wang, J., Bucher, B.: VLFM: Vision-language frontier maps for zero-shot semantic navigation. In: 2nd Workshop on Language and Robot Learning: Language as Grounding (2023)
68. Korhonen, T., Hostikka, S.: Fire dynamics simulator with evacuation: FDS+Evac (version 5). VTT Technical Research Centre of Finland (2014)
69. Ronchi, E., Nilsson, D.: Fire evacuation in high-rise buildings: a review of human behaviour and modelling research. Fire science reviews. **2**, 1–21 (2013)
70. Ronchi, E., Kuligowski, E.D., Reneke, P.A., Peacock, R.D., Nilsson, D.: The process of verification and validation of building fire evacuation models. NIST Technical Note. **1822** (2013)
71. Lovreglio, R., Kuligowski, E., Gwynne, S., Boyce, K.: A pre-evacuation database for use in egress simulations. Fire Saf. J. **105**, 107–128 (2019)
72. Helbing, D., Molnar, P.: Social force model for pedestrian dynamics. Phys. Rev. E **51**(5), 4282 (1995)
73. Chen, C.T.: Linear system theory and design (Saunders college publishing, 1984)
74. Mayne, D.Q., Kerrigan, E.C.: Tube-based robust nonlinear model predictive control. IFAC Proceedings Volumes. **40**(12), 36–41 (2007)
75. Robust model predictive control using tubes: Langson, W., Chryssochoos I., Raković, S.V., Mayne, D.Q. Automatica **40**, 125–133 (2004)
76. Robotis. TurtleBot3. https://www.robotis.us/turtlebot-3/
77. Robotis. Robotis-git/turtlebot3. https://github.com/ROBOTIS-GIT/turtlebot3
78. Baglioni, M.: Robot and victims trajectories in search-and-rescue scenario (2023). https://doi.org/10.4121/22270498
79. Tringali, A., Cocuzza, S.: Globally optimal inverse kinematics method for a redundant robot manipulator with linear and nonlinear constraints. Robotics **9**(3), 61 (2020)
80. Dhouib, S.: Hierarchical coverage repair policies optimization by Dhouib-Matrix-4 metaheuristic for wireless sensor networks using mobile robot. Int. J. Eng. **36**(12), 2153–2160 (2023)
81. Li, J., Ran, M., Wang, H., Xie, L.: MPC-based unified trajectory planning and tracking control approach for automated guided vehicles, in 2019 IEEE 15th International Conference on Control and Automation (ICCA) (IEEE, 2019), pp. 374–380
82. Brooks, A., Kaupp, T., Makarenko, A.: Randomised MPC-based motion-planning for mobile robot obstacle avoidance. In: 2009 IEEE International Conference on Robotics and Automation (IEEE, 2009), pp. 3962–3967
83. LaValle, S.M.: Planning Algorithms (Cambridge University Press, 2006)
84. Viet, H.H., Dang, V.H., Laskar, M.N.U., Chung, T.: BA*: an online complete coverage algorithm for cleaning robots. Appl. Intell. **39**(2), 217–235 (2013)
85. Choset, H., Pignon, P.: Coverage path planning: the Boustrophedon cellular decomposition. In: Zelinsky, A. (ed.) Field and Service Robotics, pp. 203–209. Springer, London (1998)

86. Wang, Y., Boyd, S.: Fast model predictive control using online optimization. IEEE Trans. Control Syst. Technol. **18**(2), 267–278 (2009)
87. Kumar, P., Anoohya, B.B., Padhi, R.: Model predictive static programming for optimal command tracking: a fast model predictive control paradigm. J. Dyn. Syst. Meas. Contr. **141**(2), 021014 (2019)
88. Diehl, M., Ferreau, H.J., Haverbeke, N.: Efficient numerical methods for nonlinear MPC and moving horizon estimation. Nonlinear model predictive control: towards new challenging applications pp. 391–417 (2009)
89. Köhler, J., Soloperto, R., Müller, M.A., Allgöwer, F.: A computationally efficient robust model predictive control framework for uncertain nonlinear systems. IEEE Trans. Autom. Control **66**(2), 794–801 (2020)
90. Boccia, A., Grüne, L., Worthmann, K.: Stability and feasibility of state constrained MPC without stabilizing terminal constraints. Systems & control letters **72**, 14–21 (2014)
91. Fang, X., Chen, W.H.: Model predictive control with preview: Recursive feasibility and stability. IEEE Control Systems Letters. **6**, 2647–2652 (2022)
92. iRobot Education. iRobot Create 3. https://edu.irobot.com/what-we-offer/create3

Mirko Baglioni received the MSc degree from University of Pisa, Italy, in 2018, after accomplishing the MSc thesis project in Örebro University, Sweden. He is currently a PhD student at the Faculty of Aerospace Engineering, Delft University of Technology, the Netherlands. His research is focused on optimization control in presence of uncertainties, including robust and stochastic model predictive control, as well as on AI-based control, with applications in search-and-rescue robotics.

Anahita Jamshidnejad received the PhD *cum laude* from the Delft University of Technology, the Netherlands, in 2017. She is currently an Assistant Professor at the Delft University of Technology. Her main research interests include optimization theory in engineering problems, integrated control methods, and real-time model predictive control, with applications to autonomous robotic systems and road traffic.