Received February 10, 2021, accepted February 23, 2021. Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2021.3063782

Hierarchical Decomposed-Objective Model Predictive Control for Autonomous Casualty Extraction

RONI PERMANA SAPUTRA[1,2], NEMANJA RAKICEVIC[1], DIGBY CHAPPELL[1,3], KE WANG[1], AND PETAR KORMUSHEV[1]

[1]Robot Intelligence Lab, Dyson School of Design Engineering, Imperial College London, London SW7 2AZ, U.K.
[2]Research Center for Electrical Power and Mechatronics, Indonesian Institute of Sciences (LIPI), Bandung 40135, Indonesia
[3]Department of Computing, Imperial College London, London SW7 2AZ, U.K.

Corresponding author: Roni Permana Saputra (e-mail: r.saputra16@imperial.ac.uk).

The work of Roni Permana Saputra was supported by Ph.D. programs through the Indonesia Endowment Fund for EducationLPDP under Grant 20160722018347. The work of Nemanja Rakicevic was supported by the President's Ph.D. Scholarship from the Imperial College London. The work of Digby Chappell was supported by the UKRI CDT in AI for Healthcare under Grant P/S023283/1. The work of Ke Wang was supported by the CSC Imperial Scholarship.

ABSTRACT In recent years, several robots have been developed and deployed to perform casualty extraction tasks. However, the majority of these robots are overly complex, and require teleoperation via either a skilled operator or a specialised device, and often the operator must be present at the scene to navigate safely around the casualty. Instead, improving the autonomy of such robots can reduce the reliance on expert operators and potentially unstable communication systems, while still extracting the casualty in a safe manner. There are several stages in the casualty extraction procedure, from navigating to the location of the emergency, safely approaching and loading the casualty, to finally navigating back to the medical assistance location. In this paper, we propose a Hierarchical Decomposed-Objective based Model Predictive Control (HiDO-MPC) method for safely approaching and manoeuvring around the casualty. We implement this controller on ResQbot — a proof-of-concept mobile rescue robot we previously developed — capable of safely rescuing an injured person lying on the ground, i.e. performing the casualty extraction procedure. HiDO-MPC achieves the desired casualty extraction behaviour by decomposing the main objective into multiple sub-objectives with a hierarchical structure. At every time step, the controller evaluates this hierarchical decomposed objective and generates the optimal control decision. We have conducted a number of experiments both in simulation and using the real robot to evaluate the proposed method's performance, and compare it with baseline approaches. The results demonstrate that the proposed control strategy gives significantly better results than baseline approaches in terms of accuracy, robustness, and execution time, when applied to casualty extraction scenarios.

INDEX TERMS Autonomous casualty extraction, mobile rescue robot, mobile robot control, model predictive control, search and rescue.

I. INTRODUCTION

A number of research studies have been conducted to develop mobile rescue robots that can perform rescue interventions, and more specifically casualty extraction tasks [1]–[8]. Some of these robots have been also deployed to assist the first responders in practice rescue mission scenarios. The existing robots that are currently in use require a high degree of human intervention, in terms of teleoperation or presence at the scene, to help the robot in performing the mission [1]. In many scenarios, teleoperation can be difficult to execute properly, for reasons such as: (i) limitations on the teleoperator's perception, e.g. a limited field of view; (ii) significant delays in signal transmission which render the teleoperation difficult; (iii) teleoperating the robot with first-person perspective requires a highly experienced operator, which limits the scalability of the number of robots deployed. Therefore, adding a certain degree of autonomy to mobile rescue robots is a crucial milestone on the road to developing a reliable

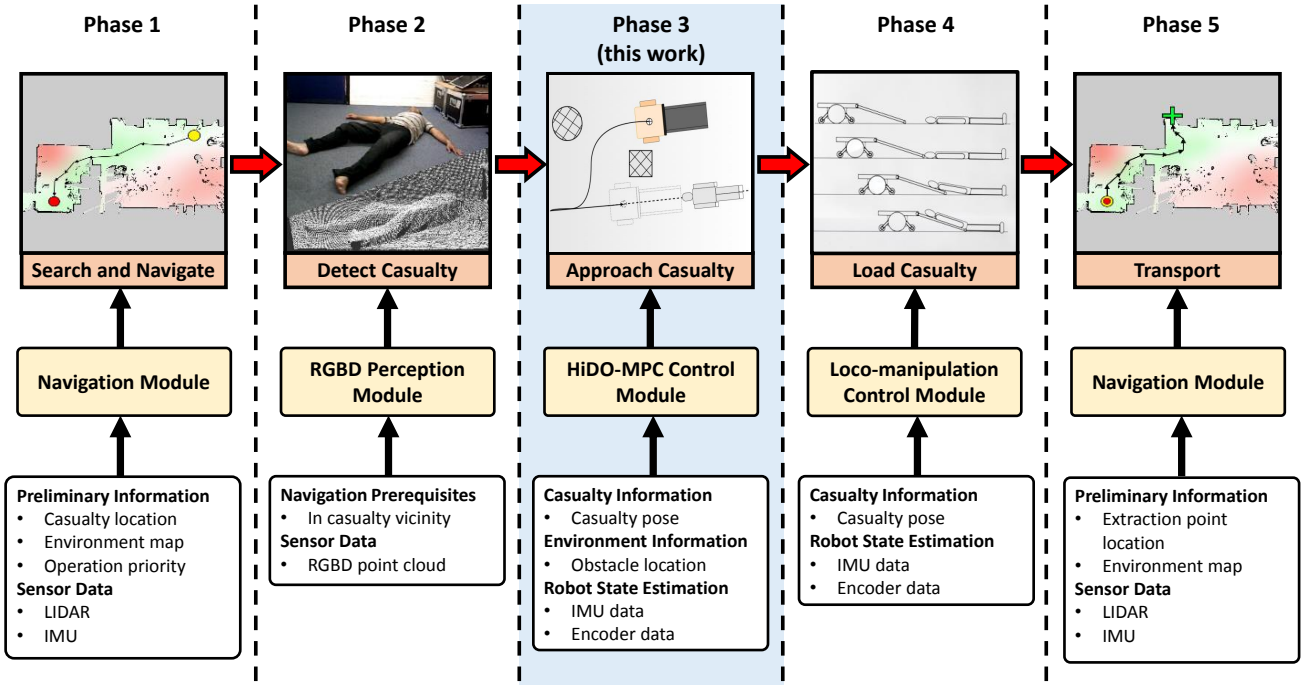


FIGURE 1. The phases that occur during casualty extraction in a search and rescue (SAR) scenario. The figure illustrates robot modules implemented on ResQbot corresponding to each of the phases. The phase highlighted in light blue is the focus of the work presented in this paper.

rescue platform for disaster response.

In the following subsections we formalise the casualty extraction procedure and additional requirements which present the basis for search and rescue (SAR) robots and control algorithm development.

A. THE CASUALTY EXTRACTION PROCEDURE

In order to formalise the casualty extraction procedure, we identify several key phases. Described in Fig. 1, each phase has distinct requirements that must be met by the corresponding robot modules, in order to execute the extraction safely and successfully. We define the five main phases of the casualty extraction procedure as follows:

Phase 1: Search and navigate. At the very beginning, the mobile rescue robot is given initial information about the task. This includes a rough estimate of all possible casualty positions, the state of their vital signs, environment map, and the feasible scenarios for extracting each of the casualties. If all the required data is well received, the robot can then navigate to the location of the casualty to start the extraction process.

Phase 2: Detect the casualty. Once the target casualty is within the range of the robot's field of view (i.e. within the perception range of the robot), the robot determines the casualty's position and orientation with respect to its own reference frame. This phase requires a robust perception module in order to provide an accurate casualty pose estimate.

Phase 3: Safely approach the casualty. After determining the casualty's pose with respects to its own frame, the robot proceeds to approaching the casualty. The robot approaches the casualty safely until reaching a desired target pose from which the casualty can be loaded. The ability of the robot to safely manoeuvre around the casualty is critical in this phase, as a misaligned approach could lead to a collision with the casualty. Similarly, the robot must be able to accurately perform the approach while avoiding obstacles and withstanding external disturbances, as these can affect the approach and compromise the safety of the loading procedure.

Phase 4: Load the casualty. When the approach phase has been safely and accurately completed, the robot can then start to load the casualty. This must be performed gently and smoothly to minimise risk of further injury to the casualty.

Phase 5: Transporting the casualty. The robot begins to transport the casualty, after ensuring that the casualty is loaded safely onto the stretcher. In this phase, the mobile robot is required to safely navigate its way from the disaster scene towards a safe place where the casualty can receive further medical assistance.

B. AUTONOMOUS CASUALTY EXTRACTION ROBOT

Our long term research goal is to develop an autonomous mobile robot that can help in emergency situations, with the ability to autonomously rescue an injured person lying on the ground — i.e. casualty extraction. The fundamental characteristics that an autonomous rescue robot performing casualty extraction should possess are (summarised in Fig. 2):

- **Safe**: It is critical that the safety of the casualty is ensured at all times. In particular, approaching, loading

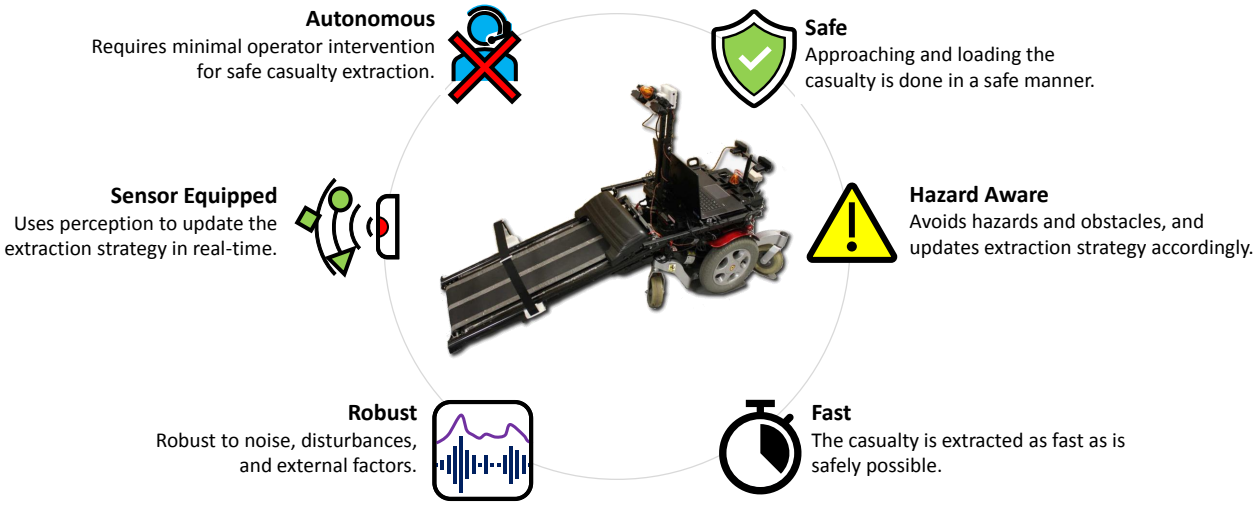


FIGURE 2. ResQbot — a previously developed robot (in the middle) for autonomous casualty extraction — and several fundamental aspects used behind the design of good casualty extraction robots.

and transporting the casualty should be performed in a way that minimises the risk of additional injury to the casualty.

- **Hazard-aware**: The casualty extraction process can be hazardous as the environment could be dynamically changing during the process. To adapt to this, the rescue robot should be able to update its extraction strategy in real-time according to current hazards and obstacles.
- **Time-efficient**: In order to preserve the casualty's health, extraction to medical professionals should be performed as fast and as safely as possible.
- **Robust**: Casualty extraction environments can be challenging for robot's sensors and actuators, so it is important that control algorithms are robust to noise, disturbances, and other external factors.
- **Sensor-equipped**: The robot should be able to sense hazards, obstacles, casualties, and reliably provide this information to the control algorithms governing the casualty extraction procedure.
- **Autonomous**: Operator communication is a common problem in challenging extraction environments, so casualty extraction should be performed autonomously after initial information is provided.

C. MAIN RESEARCH FOCUS AND MOTIVATION

In our previous works, we focused on developing methods for casualty detection (see Phase 2 in Fig. 1) [9], [10] and the procedure of safely loading the casualty (see Phase 4 in Fig. 1) [11], [12], as part of a complete casualty extraction scenario. In this study, we investigate and develop a novel control strategy in order to safely and autonomously approach the casualty prior to the casualty loading process (see Fig. 1 in Phase 3). This phase is one of the most delicate phases (along with loading the casualty), in which the robot will operate in close proximity to the casualty and the controller has to prioritise the casualty's safety as part of the controller constraints.

In our work presented in [12], we conducted a number of experiments to investigate ResQbot[1], the proposed search and rescue robot, in performing the casualty extraction procedure safely by controlling it via teleoperation. By observing how the casualty extraction procedure is safely executed via teleoperation by an experienced operator, we can derive a heuristic procedure of how the mobile robot should safely approach the target casualty. Based on our observation, the casualty-approaching procedure can be decomposed into three subsequent tasks: (i) aligning the robot position to the casualty orientation, (ii) adjusting the robot orientation with respect to the casualty orientation and (iii) gently approaching the casualty prior to the casualty loading process.

While achieving each task objective is critical for executing the proceeding task, keeping the prior task objectives in mind during every subtask execution is also essential in order to achieve the overall task goals. Therefore, the three subtasks are executed sequentially while tracking the states of the prior subtasks. We call this process 'task execution as a hierarchical decomposed-objective approach'.

In order to control ResQbot in autonomously executing the casualty-approaching task using the hierarchical decomposed-objective approach, we investigate a number of studies in the mobile robot control domain, mainly using optimal control methods. We focus on investigating optimal control methods since a large number of studies using these methods have been reported to be successful in various applications relating to autonomous mobile robots. Optimal control methods could address the limitation of reported classical methods, while offering broader applications in which either of the following challenges occurs:

- The dynamics of the controlled systems are too difficult to handle.
- The applications require many operating constraints that must be satisfied.

[1]https://www.imperial.ac.uk/robot-intelligence/robots/resqbot/

Model predictive control (MPC) – that can be derived as a finite horizon optimal control problem (OCP) – is one of the most widely used advanced control technique, and is implemented in various applications, including those of mobile robot control [13]–[22]. The ability of an MPC controller to handle a wide variety of constraints and its feasibility to be implemented in real-time applications are the reasons why this method is so popular. Nevertheless, to the best of our knowledge, most of the MPC formulations presented in these studies are designed to achieve optimal controls that produce a single control behaviour over the whole trajectory.

To execute a hierarchical decomposed-objective task — such as the autonomous casualty-approaching problem explained in the earlier paragraphs — a single control behaviour could be non-optimal for achieving the overall objectives. Therefore, the focus of our study is investigating the MPC formulation that can handle the hierarchical decomposed-objective execution approach while satisfying the operational constraints, such as the non-holonomic constraint of ResQbot and the safety constraint requirements.

D. CONTRIBUTIONS

The main contributions of this work can be summarised as follows:

1) The novel hierarchical decomposed-objective model predictive control (HiDO-MPC) approach — described in algorithm 1 — to solve tasks with decomposed objectives, such as the casualty-approaching task described in the subsection IV-A.
2) The formulation of the proposed HiDO-MPC for the casualty-approaching task, including the hierarchical objective function formulation and the required constraints formulation.
3) The performance evaluation of the proposed method in the autonomous casualty-approaching scenario, including task accomplishment and safety performance.
4) The source code implementation of the proposed method used in the experiments that will be publicly available on the project website[2] upon publication.

This paper is further organised as follows. In Section II we review the literature related to autonomous mobile rescue robots, including casualty-extraction mobile robots and autonomous control for mobile robots. In Section III we define the casualty approach problem. We present our proposed approach and the algorithm of HiDO-MPC in Section IV, including the detail-controller design formulation of the proposed method, the computational implementation of the control problem, and the introduction of the other three methods — adapted from the state of the art — to compare them to our proposed method. The experimental set-up is explained in Section V, and the discussion of the results is provided in Section VI. Finally, we conclude our findings in Section VII.

[2]https://sites.google.com/view/hido-mpc-resqbot/source-code

II. RELATED WORK

In this section we present an overview of the existing mobile rescue robots designed for casualty extraction and existing control methods that can be implemented for autonomous mobile robot control.

A. RESCUE ROBOT DESIGN AND CONTROL

One of the design streams for casualty extraction robots is a robot with a humanoid upper body form design, with a heavy-duty dual-arm manipulator. Battlefield Extraction Assist Robot (BEAR) developed by Vecna Technologies [1], [2] and HERCULES robot developed by the Agency for Defense Development, Republic of Korea [3] are examples of a semi-humanoid form mobile robot platforms, designed and developed specifically for casualty-extraction procedures. These robots are designed to be able to perform casualty extraction using their arms by scooping, lifting up and carrying the casualty (see presented work in [3]). While this casualty extraction procedure seems to be flexible, feasible and mimics how a normal person handles a casualty, controlling such a complex robot performing an intricate and sensitive task is a significant challenge. Teleoperating such a complex system most likely requires more than one highly skilled and experienced operator. Developing a fully autonomous controller for such a system is a non-trivial task.

On the other hand, another design stream has been proposed in several research studies [4]–[8]. In these studies, a stretcher-type construction or litter is implemented to achieve a more compact and simpler mechanism, compared to the semi-humanoid-type design. In the stretcher-type design, the casualty extraction procedure is achieved by using a conveyor belt mechanism that loads a casualty from the ground onto the mobile platform without any direct lifting process. The conveyor belt module properly supports the casualty's body during the process so it can additionally serve as a stretcher bed for transport.

Introduced in our previous works, ResQbot (see Fig. 2) is a stretcher-type mobile rescue robot designed to load the casualty using a loco-manipulation approach, i.e. using the robot's locomotion system — wheeled locomotion — to perform a manipulation task [11]. It was found that teleoperating the loco-manipulation process allows the robot to load the casualty while ensuring key safety thresholds are adhered to [12]; avoiding possible causes of head or neck injury [23], [24].

Compared to the semi-humanoid-type design, the stretcher-type mechanism is significantly simpler. Therefore, adding certain degree of autonomy to the stretcher-type robots is more feasible than to the semi-humanoid-type robots.

B. MOBILE ROBOT CONTROL

A multitude of research studies have been conducted in the field of autonomous mobile robot control, especially for non-holonomic mobile robot systems [25]–[30]. In general, these studies address proposed solutions for three main control

problems, including point stabilisation, trajectory tracking and path following. A wide range of techniques have been proposed, including back-stepping [26], dynamic feedback linearisation [27], sliding mode control [28] and Lyapunov control [29]. However, according to recent studies [30], it is observed that most of these techniques are not designed to be able to cope with a variety of constraints. These constraints include both physical and behavioural constraints of the robot, such as a complex state space and the requirements for safe and efficient executions [31].

C. MODEL PREDICTIVE CONTROL

A large number of studies have been reported in the domain of optimal control methods as they prove to be successful in a range of applications. The methods are widely used, especially in applications in which the dynamics of the controlled systems are too difficult to handle or the applications require many operating constraints to be satisfied [31]–[36].

Meanwhile, MPC — a feedback control technique that solves optimal control problems over a finite time horizon — has recently gained popularity. Due to its flexibility to handle constraints and the capacity for real-time computation, MPC has been implemented in various applications, including autonomous mobile robot control [13]–[22]. A large number of studies have been conducted using variations of MPC approaches and formulations for solving non-holonomic mobile robot controls that cover three general applications: point stabilisation, trajectory tracking, and collision avoidance [15]–[22]. The work presented by Neunert et al. in [20], for instance, proposes a framework for real-time non-linear MPC that solves mobile robot trajectory optimisation and tracking problems simultaneously in a single approach. The framework introduces an iterative optimal control algorithm, a.k.a. sequential linear quadratic, to solve the non-linear MPC problem. Li et al., in [17], propose using a primal-dual neural network to solve the quadratic programming (QP) problem in a finite receding horizon to achieve trajectory control of mobile robot systems. The proposed approach is designed to make the formulated constrained QP cost function converge to the exact optimal values and achieve real-time performance on a real mobile robot system. Another work that incorporates neural networks into the MPC scheme is presented by Hirose et al. in [19]. In contrast to the work in [17], the authors in [19] use a deep neural network to learn the MPC policy.

Despite a wide range of problems presented in the MPC studies associated with mobile robot controls, the presented MPC formulations are designed to achieve optimal controls using a single objective function and produce a single control behaviour over the whole trajectory. It is possible that single control objectives could be non-optimal for achieving more complex tasks, such as those that require multiple control behaviours along a single trajectory. This means that the task could require a different control behaviour (i.e. control strategy) in different state-space regions to achieve the desired overall performance.

D. MULTI-OBJECTIVE AND HIERARCHICAL CONTROL

In order to solve high-dimensional problems such as complex multi-behaviour tasks and multi-constrained systems, several studies have been proposed using hierarchical and multi-objective controller approaches. These approaches have been implemented successfully in a wide range of applications [37]–[42].

Multi-objective model predictive controls have been proposed in a number of studies in order to achieve desired performances [37]–[40]. In [38], for example, a weighted sum of multiple objective functions was proposed, and in [39] the authors compute a Pareto optimal solution to solve the multi-objective MPC as a multi-parametric multi-objective linear or quadratic program. A study on MPC using a more complex objective function has been also presented by Zhanget et al. in [40]. In this study, a complex objective problem is decomposed into a sequential-multi-objective MPC formulation. This multi-objective approach is evaluated sequentially based on priority, in contrast to the classical multi-objective MPC strategies with weighting factors.

On the other hand, various hierarchical approaches have been proposed based on several interests, including the resolution relevance of information, task decomposition, behaviours and hierarchical multi-objectives. The study presented in [43], for instance, proposes a hierarchical structure of mobile robot controllers based on information surrounding the resolution relevance of the environment. The highest level of path-planning is generated based on a coarse and incomplete world description, while the lowest level controller will refine the robot motion based on the more-detailed obtained information, during the course. Similarly, the work proposed by Moore and Flann in [44] is characterised by a hierarchical task-decomposition approach. The hierarchical structure of the decomposed tasks in this work is similar to the structure in [43], including the high-level planner, trajectory generator and low-level actuator controller.

In contrast, hierarchical controllers based on behaviour approaches are proposed in [45]–[50]. In general, the purpose of these studies is to generate a complex behaviour by decomposing the whole complex behaviour into several simpler behaviours in a hierarchical structure. Each behaviour is controlled independently in combination with the others. In the case of mobile robot controls, these behaviours could be path-following, obstacle-avoidance, wall-following, goal-reaching or emergency-related.

The problem becomes how to combine or formulate these behaviours, which might conflict with each other, and produce the desired whole complex behaviour. Hasegawa and Fukuda in [45] propose a learning method (i.e. multiple regression analysis) to determine the deficient sub-controller (i.e. sub-behaviour) in the system. Meanwhile, Abdessemed et al., in [49], combine behaviour controllers by formulating them into a set of fuzzy-rule statements, so that the problem become a fuzzy decision-making one. Similarly, in [48], Meléndez and Castillo also convert the problem into a fuzzy decision-making one, in which they propose to use the

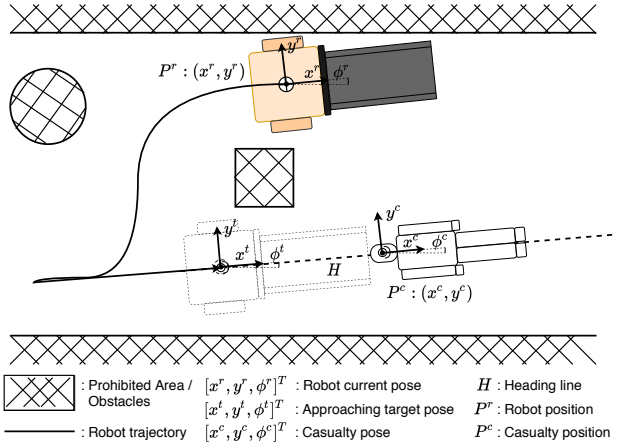


FIGURE 3. Problem formulation. This study is focused on the proposed method for controlling ResQbot to safely approach the casualty, with constrained manoeuvrability.

weighted fuzzy inference system as it is presented in [47].

Multi-objective MPC strategies employing hierarchical controllers for high-level planning and low-level control have been introduced in several works on autonomous mobile robot controls [51]–[53]. Similar to the structure presented in [44], the high-level planner generates a collision-free trajectory computed online to reach a desired target. The low-level controller is then responsible for the robot's trajectory tracking by controlling the actuators, such as the robot's steering angle. Another study introduces a hierarchical structure to multi-robot controllers in order to solve a complex MPC problem [54]. It is applied to control multiple mobile robots wherein the high-level component optimises collective robot configurations while the low-level component performs obstacle avoidance.

In this study we derive a heuristic procedure for the casualty-approaching task as a hierarchical decomposed-objective approach in order to achieve a specific behaviour (see IV-A). We formulate the problem as a hierarchical control task based on a multiple-behaviour approach and propose HiDO-MPC to solve the problem. Even though in this study we focus on the implementation for solving the autonomous casualty-approaching task, the proposed HiDO-MPC approach could also have wider practical applications. In particular this approach would be suitable for complex tasks that can be decomposed into several subsequent tasks, executed using the hierarchical decomposed-objective approach. For instance, solving complex manipulation tasks as presented in [55], which is using a similar high-level concept (i.e. hierarchical task decomposition).

III. PROBLEM FORMULATION

A. CASUALTY APPROACH PROBLEM

The casualty approach problem — illustrated in Fig. 3 — can be formulated as an optimal control problem, where the high-level objective is to minimise the state deviation from the robot's current pose, $\mathbf{x}^r$, to the robot's target pose, $\mathbf{x}^t$ corresponding to the casualty's pose, $\mathbf{x}^c$:

$$J = ||\mathbf{x}^r - \mathbf{x}^t||^2. \tag{1}$$

The casualty extraction robot, ResQbot, is a non-holonomic robot with position and orientation $\mathbf{x}^r = [x^r, y^r, \phi^r]^\intercal$. Similarly, the casualty has 2D pose $\mathbf{x}^c = [x^c, y^c, \phi^c]^\intercal$. ResQbot is controlled with input $\mathbf{u} = [v, \omega]^\intercal$ where v and ω are forward velocity and angular velocity, respectively. The robot dynamics are therefore:

$$\dot{\mathbf{x}}^r(t) = \mathbf{f}\big(\mathbf{x}^r(t),\, \mathbf{u}(t)\big) = \begin{bmatrix} v(t)\cos\phi(t) \\ v(t)\sin\phi(t) \\ \omega(t) \end{bmatrix}, \tag{2}$$

or, in discrete time:

$$\begin{aligned} \mathbf{x}^r_{|k+1} &= \mathbf{f}(\mathbf{x}^r_{|k}, \mathbf{u}_{|k}) \\ &= \underbrace{\mathbf{x}^r_{|k}}_{\text{current state}} + \Delta t \underbrace{\begin{bmatrix} v_{|k}\cos\phi^r_{|k} \\ v_{|k}\sin\phi^r_{|k} \\ \omega_{|k} \end{bmatrix}}_{\text{state transition}}. \end{aligned} \tag{3}$$

To ensure safety during the casualty approach, the robot needs to avoid colliding with all possible obstacles (i.e. prohibited area in Fig. 3) in all directions, so that:

$$\mathbf{x}^r_{|k} \in X^{free} \tag{4}$$

where X^{free} is the set by all possible robot states in which the robot is free from collision, including collision with the casualty.

The casualty approach problem can therefore be written as a model predictive control problem as follows:

$$\begin{aligned} \min_{\mathbf{u}} \;\; & J_N(\mathbf{x}_0, \mathbf{u}) = \sum_{k=0}^{N-1} \left\|\mathbf{x}^r_{|k} - \mathbf{x}^t\right\|^2_{\mathbf{Q}} + \left\|\mathbf{u}_{|k} - \mathbf{u}^t\right\|^2_{\mathbf{R}} \\ \text{s.t.} \;\; & \mathbf{x}^r_{|k+1} = \mathbf{f}(\mathbf{x}^r_{|k}, \mathbf{u}_{|k}), \;\; \forall k \in [0, N-1] \\ & \mathbf{x}^r_{|0} = \mathbf{x}_0 \\ & \mathbf{u}_{|k} \in U, \;\; \forall k \in [0, N-1] \\ & \mathbf{x}^r_{|k} \in X^{free}, \;\; \forall k \in [0, N] \end{aligned} \tag{5}$$

where $\left\|\mathbf{x}^r_{|k} - \mathbf{x}^t\right\|^2_{\mathbf{Q}}$ and $\left\|\mathbf{u}_{|k} - \mathbf{u}^t\right\|^2_{\mathbf{R}}$ are the functions of the state deviation and the control effort, respectively. The expression $\|\mathbf{A}\|^2_{\mathbf{B}} \equiv \mathbf{A}^T\mathbf{B}\mathbf{A}$. The matrices $\mathbf{Q}$, $\mathbf{R}$, and $\mathbf{P}$ are positive definite symmetric weighting-matrices of the appropriate dimensions.

B. CONTROLLER DESIGN REQUIREMENTS

There are two main design requirements that the designed controller needs to achieve:

- **Task accomplishment**: the controller generates decision control that enables the ResQbot to execute the casualty approaching task and to achieve the desired behaviour described in subsection IV-A. We formulate a

specific hierarchical cost function — derived in subsection IV-C — for the proposed HiDO-MPC, to achieve this task accomplishment requirement.
- **Collision free operation**: During the task execution, the robot needs to avoid colliding with all possible obstacles (i.e. prohibited area in Fig. 3) in all directions, including walls and the casualty. To achieve this requirement, we formulate the collision avoidance constraint function — derived in subsection IV-D — and include it as one of the HiDO-MPC constraints.

C. SCOPE AND ASSUMPTIONS

In this paper, we focus on the development of an MPC-based controller to generate control commands for the ResQbot to safely approach the target casualty as a part of a complete casualty extraction scenario. The controller takes the state of the current robot $\mathbf{x}^r$, casualty $\mathbf{x}^c$, and the desired robot state $\mathbf{x}^t$ as input for controlling the robot motion, as well as the environmental information, such as the map and the list of obstacles.

Conceptually, all the information is given in real-time by a state observer module. This state observer module is not within the scope of the current study. In simulation experiments all the information is available to the controller. In real robot experiments, the real-time robot states are provided by the implementation of simultaneous localisation and mapping (SLAM), presented in [56], [57], which is also out of scope of this paper.

Another assumption in this study is that the controller developed in this work would generate control commands for ResQbot, including linear velocity command v and angular velocity ω. The low-level controller — actuating the robot actuators based on the control commands v and ω — is also beyond the scope of this work.

IV. PROPOSED METHOD

In this study, we propose a hierarchical decomposed-objective model predictive control approach (HiDO-MPC) to achieve the overall objective of the casualty approach task (see III-A). The proposed approach is derived based on our observation of how the casualty extraction procedure is safely executed via teleoperation by an experienced operator in [12].

A. CASUALTY APPROACHING HIERARCHICAL TASK DECOMPOSITION

To achieve the overall objective of the casualty approach task, we decompose the task of approaching the casualty into three subtasks (illustrated in Fig. 4):

1) aligning the robot position to the casualty heading line (H) until the desired threshold;
2) adjusting the robot orientation ϕ^r with respect to the casualty orientation ϕ^r up to the desired threshold $\Delta\phi$, while keeping the robot alignment; and
3) approaching the casualty, while keeping the robot alignment and the robot orientation.

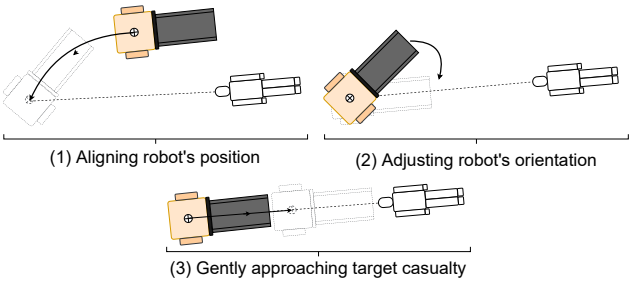


FIGURE 4. Casualty approaching task decomposition inspired by the loco-manipulation approach presented in [11].

These three subtasks are executed sequentially while tracking the states of the prior subtasks to achieve the overall task goals. In contrast to a pure sequential execution, we called this execution as a hierarchical execution.

We introduce three specific cost functions to the desired behaviours that correspond to the three subtasks during the task execution:

- **Distance-to-line objective**: Given the robot position $\mathbf{p}^r = [x^r, y^r]^\intercal$ as a spatial component of the robot states $\mathbf{x}^r$ and the finite heading line segment H as a function of the casualty pose $\mathbf{x}^c$, minimising the distance from $\mathbf{p}^r$ to the line segment H ensures the robot-casualty alignment:

$$F_1(\mathbf{x}^r, \mathbf{x}^c) = d(\mathbf{p}^r, H). \tag{6}$$

  Detailed derivation of this objective function can be found in Appendix A.
- **Heading objective**: Given the robot orientation ϕ^r and the casualty orientation ϕ^c, minimising the difference between these orientations Δ ensures that the robot could approach the casualty in the correct heading:

$$F_2(\mathbf{x}^r, \mathbf{x}^c) = \Delta(\phi^r, \phi^t) = \pi - \left|\left|\phi^r - \phi^t\right| - \pi\right|, \tag{7}$$

  where:

$$\phi \in [-\pi, \pi].$$

  so that it takes into account the periodicity of angles (i.e. angle wrapping).
- **Distance-to-point objective**: Given the current robot position $\mathbf{p}^r = [x^r, y^r]^\intercal$ and the target robot position $\mathbf{p}^t = [x^t, y^t]^\intercal$, which is a function of the casualty pose $\mathbf{x}^c$, minimising the distance from the current robot position to the target position will drive the robot to achieve the final goal of the casualty approaching task. The distance between these two points in 2D space can be defined as the Euclidean norm:

$$F_3(\mathbf{x}^r, \mathbf{x}^c) = \left\|\mathbf{p}^r - \mathbf{p}^t\right\|^2 \tag{8}$$

B. HIERARCHICAL DECOMPOSED-OBJECTIVE MPC

Fig. 5 illustrates a high-level overview of the proposed approach. We formulate HiDO-MPC to achieve the casualty approaching hierarchical task decomposition. In contrast to

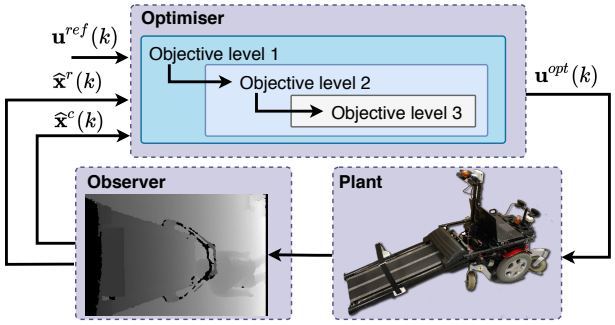


FIGURE 5. Illustration of a three-level hierarchical decomposed-objective model predictive control (HiDO-MPC) applied to the ResQbot control for autonomous casualty approach.

the generic MPC formulation, which uses a single overall objective function J_N (Eq. 5), in HiDO-MPC, we formulate decomposed objective functions corresponding to the subtasks derived in IV-A. These objective functions are then executed hierarchically to ensure that each sub-objective is achieved while maintaining the the states of the prior sub-objective.

Let us denote the stage s as a subtask of a complete overall task, $s \in S = \{1, 2, 3\}$, of the hierarchical task decomposition. Each stage $s(\cdot)$ is evaluated based on the current state $\mathbf{x}^r$ with respect to the overall goal state $\mathbf{x}^t$ and the objective of each subtask. The discrete HiDO-MPC controller here is defined as an OCP with a finite control horizon N, which evaluates the robot stage s at every sampling instant k.

The stage evaluation process determines the objective function corresponding to that stage, J_N^s. Then, the optimal control, $\mathbf{u}^*$, for the robot is produced at every time step by solving the OCP with respect to the decomposed-objective function at this respective stage, J_N^s, which satisfies the optimal value, $V_N^s(\hat{\mathbf{x}})$ (i.e. minimising the output of the cost function, J_N^s, s.t. constraints). The first element of the produced optimal control trajectory, $\mathbf{u}_0^*$, is then applied to the system. Algorithm 1 summarises the proposed approach.

C. OBJECTIVE FUNCTION DESIGN AND IMPLEMENTATION TO HIDO-MPC

We implement the three objective functions formulated in Eq. 6, 7, and 8 into the proposed HiDO-MPC algorithm (see Algorithm 1) as a three-stage hierarchical objective, $s \in S = \{1, 2, 3\}$. Each stage corresponds to the task decomposition described in subsection IV-A. We formulate the objective function at each stage, J_N^s, as a weighted combination of the three proposed objective functions with different weights $(W_1^H - W_9^H)$, written as:

- **Objective Stage 1**

$$J_N^1 = \sum_{k=0}^{N-1} W_1^H F_1(\mathbf{x}_{|k}^r, \mathbf{x}^c) + W_2^H \left\|\mathbf{u}_{|k} - \mathbf{u}^t\right\|_{\mathbf{R}}^2 \quad (9)$$

Algorithm 1: HiDO-MPC control approach for casualty approach task

Initialisation:
StageNumber := n
$s := 1 \in S = \{1, 2, 3\}$
MPCInit:
ControlHorizon := N
Define the initial robot state:
$\mathbf{x}_{|0}^r := x_0 \in X^{n_x}$
Get the target state:
$\hat{\mathbf{x}}^t \leftarrow$ StateObserver
Define the initial control: $\mathbf{u}_{|0}$
Apply $\mathbf{u}_{|0}$ to the system.
for every sampling instant $k = 1, 2, \ldots$ **do**
Estimate the states $\mathbf{x}_{|k}^r$ and $\mathbf{x}^t$:
$[\hat{\mathbf{x}}_{|k}^r, \hat{\mathbf{x}}^t] \leftarrow$ StateObserver
Evaluate $s = S(\mathbf{x}_{|k}^r, \mathbf{x}^t)$
$\forall\ s \in \{0, 1, 3\}$
Evaluate the sub-objective switching
w.r.t. current stage s:
$J_N = J_N^s$
Solve OCP w.r.t. current stage s:
Find the optimal control horizon
$\mathbf{u}^* = \{\mathbf{u}_{|0}^*, \cdots, \mathbf{u}_{|N-1}^*\} \in U^N$,
which satisfies
$J_N^s(\hat{\mathbf{x}}, \mathbf{u}^*) = V_N^s(\hat{\mathbf{x}})$.
s.t. constraints
Apply $\mathbf{u}_{|0}^*$ to the system.

- **Objective Stage 2**

$$J_N^2 = \sum_{k=0}^{N-1} W_3^H F_1(\mathbf{x}_{|k}^r, \mathbf{x}^c) + W_4^H F_2(\mathbf{x}_{|k}^r, \mathbf{x}^c) + W_5^H \left\|\mathbf{u}_{|k} - \mathbf{u}^t\right\|_{\mathbf{R}}^2 \quad (10)$$

- **Objective Stage 3**

$$J_N^3 = \sum_{k=0}^{N-1} W_6^H F_1(\mathbf{x}_{|k}^r, \mathbf{x}^c) + W_7^H F_2(\mathbf{x}_{|k}^r, \mathbf{x}^c) + W_8^H F_3(\mathbf{x}_{|k}^r, \mathbf{x}^c)) + W_9^H \left\|\mathbf{u}_{|k} - \mathbf{u}^t\right\|_{\mathbf{R}}^2 \quad (11)$$

D. COLLISION AVOIDANCE AS EQUALITY CONSTRAINTS

Another design requirement of the proposed controller is to accomplish the casualty approaching task safely.

To achieve this design requirement, we formulate the collision avoidance as one of the constraints included in the MPC formulation (Eq. 5). Inspired by the work presented in [58]–[60], we introduce collision function between two objects, written as:

$$[col]_+ = \max\{0, col\}. \quad (12)$$

We formulate the collision function, col as an expression so that the value is positive when the objects are in collision and

negative when they are collision free. We model the collision geometry of each object — including robot, casualty, and obstacles — as a set of circles, so that the optimisation process still feasible in real-time. Full derivation of the collision function used in this work can be found in Appendix B.

E. SOLVING THE OPTIMAL CONTROL PROBLEM (OCP) VIA NON-LINEAR PROGRAMMING USING CASADI

To solve the finite OCP problem in the proposed HiDO-MPC, in this study, we use `CasADi` API [61] computing the real-time optimisation problem. The OCP is computed via the API's Non-Linear Programming (NLP) using *ipopt* solver. The standard NLP formulation for the numerical parametric optimisation problem is as follows:

$$\begin{aligned} \min_{\mathbf{w}} &: \Phi(\mathbf{w}, \mathbf{p}) \\ \text{subject to: } & \mathbf{g}_1(\mathbf{w}, \mathbf{p}) \leq 0, \\ & \mathbf{g}_2(\mathbf{w}, \mathbf{p}) = 0. \end{aligned} \tag{13}$$

in which $\mathbf{w} \in \mathbb{R}^{n_w}$ is the decision variable and $\mathbf{p} \in \mathbb{R}^{n_p}$ is a known parameter vector. $\Phi(\mathbf{w}, \mathbf{p})$ is the objective function of the optimisation problem, while terms $\mathbf{g}_1(\mathbf{w}, \mathbf{p}) \leq 0$ and $\mathbf{g}_2(\mathbf{w}, \mathbf{p}) = 0$ are inequality and equality constraint functions, respectively.

The detailed derivation of transforming the proposed HiDO-MPC OCP into NLP problem can be found in Appendix C.

F. COMPARISON TO THE STATE OF THE ART

To evaluate the proposed HiDO-MPC approach, this study compared the proposed approach to several other methods, adapted from the related work. These methods include: i) sequential MPC (SMPC) adapted from [40], ii) bundled objectives MPC (BMPC) adapted from [38], and iii) vanilla MPC (VMPC) which is a generic MPC method implemented on mobile robot control as presented in [17]–[19].

The derivation of the objective function implementation of these three methods, in contrast to the HiDO-MPC can be found in Appendix D.

V. EXPERIMENTAL SETUP

To evaluate the proposed HiDO-MPC, we design several experimental scenarios, both in simulation and on the ResQbot physical robot platform. Two main controller design requirements are evaluated, the task accomplishment and collision-free operation (see III-B). Our main hypothesis is that the proposed HiDO-MPC controller could generate the desired behaviour of ResQbot (see IV-A) to safely accomplish the casualty approaching task as part of the casualty extraction procedure described in Section 1.

A. EXPERIMENTAL SCENARIOS

We have designed several experiments to emphasise each of the design requirement points. The following experiments are conducted in simulation and on the real robot:

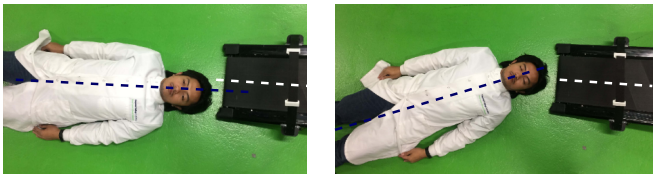


FIGURE 6. Illustration of pre-loading misalignment and heading error which lead to failure during loading and could cause additional injury to the casualty.

1) **Different obstacle densities (simulation):** This experiment is designed to show hazard-awareness capabilities by avoiding obstacles and achieving the target pose safely. This evaluates the collision-free requirement formulated as a collision avoidance constraint in the controller design.
2) **State estimation error and control perturbation (simulation):** This experiment is intended to demonstrate the robustness of the proposed controller in the presence of different types of sensor measurement errors and control disturbances. This evaluates the advantage of the implementation of the hierarchical decomposed-objective approach in HiDO-MPC (hierarchical structure) in comparison to the sequential approach methods in SMPC (sequential structure), as illustrated in Fig. 14 Appendix D.
3) **Execution time (simulation):** This experiment is designed to evaluate the time-efficiency of the controllers during task execution. This assesses the objective function formulation. Shorter execution time means that the controller could generate optimal decision control for the ResQbot to accomplish the desired task.
4) **Narrow corridor case (simulation and real):** This experiment is a case study of ResQbot performing the casualty extraction task in an environment with limited space for manoeuvring. This appraises the formulation of the ResQbot model (see III-A) implemented in real robot experiments and the robustness of the proposed controller in the presence of the actual model and measurement error.

We describe in more detail the setup of the proposed experiments in Appendix E.

B. PERFORMANCE METRICS

To quantitatively assess the task accomplishment of the proposed method, the experimental results are evaluated based on two metrics:

- Distance to heading line (i.e. misalignment) $[m]$
- Heading error $[^\circ]$

If the errors exceed 0.1 m or 10°, respectively, the experiment is considered to have failed. The reason is that if the robot were to start loading the casualty from such a pose, with large errors, it could cause significant injuries to the casualty. Fig. 6 illustrates the misalignment and heading errors between ResQbot and the casualty, before the casualty loading stage.

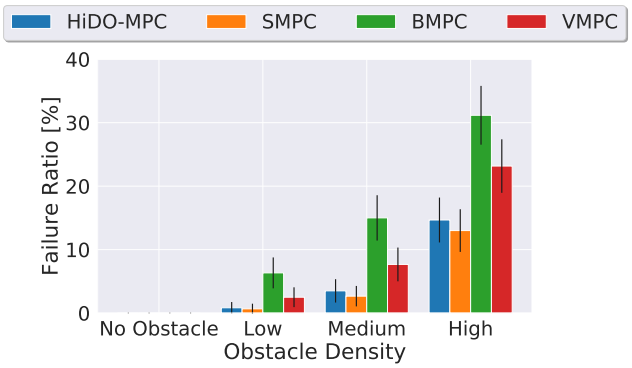


FIGURE 7. Performance comparison of the evaluated controllers in different obstacle density experiments. The failure ratio is calculated over 600 trials, and the variance is obtained based on binomial distribution.

We also calculate failure ratio to assess the HiDO-MPC performance compared to the other methods. For each set of experiments, the failure ratio is calculated as the number of failure cases (i.e. misalignment or heading error exceeds the thresholds) divided by the total number of experiments in the corresponding set.

C. HYPER-PARAMETER TUNING

Prior to the main experiments, we perform hyper-parameter tuning, including finding the optimal weights for cost functions in all MPC methods (see IV-C and Appendix D). We also observe the computation times of all controllers with respect to the prediction horizon lengths N, in order to assure that the controllers can be implemented in real-time. The results of these parameter tuning can be found in the Appendix G.

D. PHYSICAL ROBOT EXPERIMENTS

The physical robot experiments are conducted using the ResQbot platform [12] in an indoor laboratory environment. We run a set of experiments having the robot and casualty placed at pre-defined poses within the environment map, as indicated in Table 3 Appendix E-D.

To update the robot states in real-time, we use the implementation of simultaneous localisation and mapping (SLAM), presented in [56], [57]. To execute the control command from the MPC controllers, ResQbot uses its on-board low-level controller to manage the linear speed and angular rate of the differential drive wheels.

In the experiments, there are at least two sources of stochasticity contributing to the controller. First, the state estimation error produced by the SLAM implementation using 2D light detection and ranging (LIDAR) sensor. The second sources of stochasticity, is the model discrepancy of the robot dynamics model. The error from the low-level controller execution and the actual non-uniform distribution of friction (e.g. between the robot's wheels and the surface, wheel) are such instances of the source of the model discrepancy.

VI. RESULTS AND DISCUSSION

In this section, we present the results of the experimental setups described in the previous section. We further discuss how these findings relate to the controller design requirements presented in III-B.

A. THE EFFECT OF OBSTACLE DENSITY

In Fig. 7 we present the averaged failure ratio for each obstacle density (low, medium and high) and controller method, calculated over 600 trials. The overall results illustrate that while all controllers can achieve 100% successful performance in no-obstacle experiments, when obstacles are present HiDO-MPC and SMPC demonstrate significantly better performance in comparison to VMPC and BMPC, with lower average failure ratios and smaller variances. This similarity between HiDO-MPC and SMPC can be explained by the fact that both HiDO-MPC and SMPC use multiple decomposed-objective functions, although implemented differently (see Section IV-F and Fig. 14 in Appendix D). In contrast to HiDO-MPC and SMPC, BMPC and VMPC use a single fixed-objective function to complete the task. As expected, the failure ratio increases with the increasing obstacle density.

This finding demonstrates that using multiple decomposed-objective functions (implemented in HiDO-MPC and SMPC) results significantly better performance than using single objective function (implemented in BMPC and VMPC), in achieving desired casualty approaching task with collision free constraint.

B. THE EFFECT OF STATE ESTIMATION ERROR AND CONTROL PERTURBATION

In this experiment, we analyse the effect of different modelling errors (sensor measurement errors and control perturbations), on the final controller performances. The detailed detailed noise setup is described in Table 2 Appendix E.

The results of these experiments are presented in Fig. 8. The overall results shows that the presence of measurement errors and control perturbation has a significant impact on the controller performances. This especially holds for SMPC, which is in contrast to our previous findings in experiments without noise effects. We can see that even in the low obstacle density experiments, SMPC exhibits in poor performance. Based on these findings, we can conclude that in some instances the presence of modelling errors (i.e. measurement and control) has a significant effect on the SMPC performance.

Conversely, HiDO-MPC demonstrates a consistently higher performance compared to other controllers, in both the experiments with and without modelling errors. The discrepancy between HiDO-MPC and SMPC performances could be explained by the difference in implementation of the decomposed-objective functions (see IV-F and Fig. 14 in Appendix D). In SMPC the multiple decomposed-objective functions are implemented as an open-loop sequential structure. Once a subtask objective is achieved, this objective

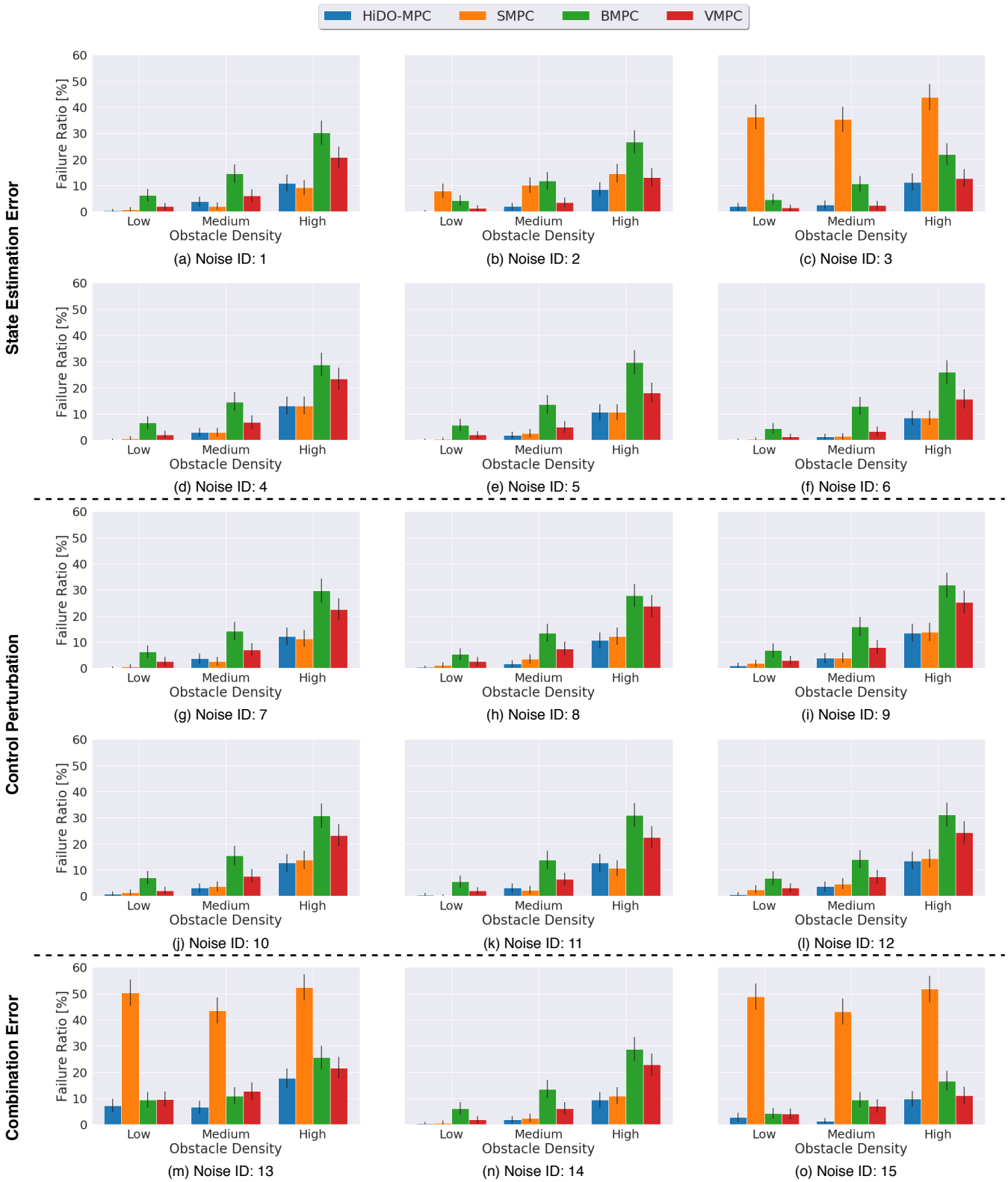


FIGURE 8. **Controller performance comparison under 15 different experimental setups involving noise effects, including state estimation errors and control perturbation (see the detailed noise setup in Table 2 Appendix E). Each graph shows the performance with: $[a-c]$: position (x, y) estimation errors; $[d-f]$: orientation ϕ estimation error; $[g-i]$: linear velocity v control signal perturbation; $[j-l]$: angular velocity ω control signal perturbation; $[m]$: both position and orientation estimation errors; $[n]$: both linear and angular velocity control signal perturbations; $[o]$: combination of all noise effects.**

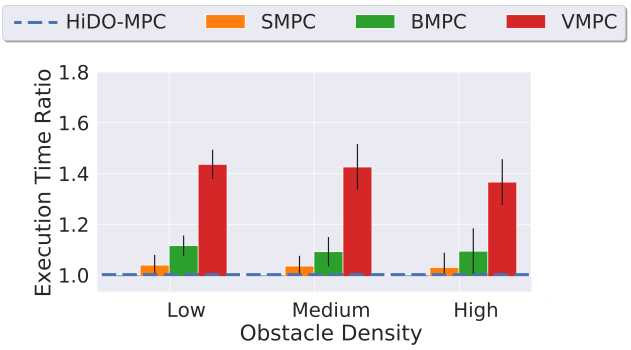


FIGURE 9. Execution time of SMPC, BMPC, and VMPC controllers represented as a ratio with respect to the HIDO-MPC execution time.

will not be maintained once the next sequence objective is initiated. In contrast, HiDO-MPC uses closed-loop hierarchical structure, in which each decomposed-objective function is evaluated hierarchically at every iteration, and the current stage objective is executed while maintaining the former subsequent objectives. This specific implementation difference between HiDO-MPC and SMPC methods makes the former method results much more robust performance in the presence of model error.

The graphs presented in Fig. 8-a, 8-b, and 8-c show the controller failure ratios in the experiments with robot position estimation error and the effect of robot orientation estimation errors to the controller failure ratios are illustrated in Fig. 8-d, 8-e, and 8-f. We can observe from the graphs that the position estimation errors have a significant effect on the SMPC failure ratio, as a higher position estimation error results in a higher SMPC failure ratio. In contrast to the robot position estimation errors, the results show that robot orientation errors have a less significant effect on the controller performances (i.e. failure ratio). This finding is explained by the fact that ResQbot model has non-holonomic constraints. As a consequence of these constraints, it is easier to adjust the robot's heading only, as opposed to adjusting the position errors.

Fig. (8-g – 8-l) show the experiment results in which control perturbations — in linear velocity (g-i) and angular velocity (j-l) — are introduced to the system. The figures show that control perturbations have a small effect on the controller performances, and the effect is not as significant as the effect of the robot position estimation errors.

The failure ratio results from the experiments combining all the noise factors, including state estimation errors and control perturbations, is presented in Fig. 8-m, 8-n, 8-o. These finding highlight the fact that the proposed HiDO-MPC approach is more robust to the presence of noise and disturbance while achieving desired casualty approaching task.

C. EXECUTION TIME COMPARISON

Fig. 9 shows the ratio of the SMPC, BMPC, and VMPC execution times with respect to the HiDO-MPC execution times. These results are obtained from the successful subset of all the conducted experiments (i.e. all the trials in which the target was reached without failure) considering all obstacle densities both with and without noise effects. The figures show that in general HiDO-MPC requires less time to execute the task compared to the other controllers. In contrast, VMPC requires a significant amount of time to reach the steady-state error, because this controller is sub-optimal and produces more oscillations.

These findings demonstrate that the proposed HiDO-MPC outperform the other controllers in generating optimal trajectory for the desired casualty approaching task.

D. NARROW CORRIDOR EXPERIMENT

In this experiment, the task is to approach the casualty safely in the environment where the robot's manoeuvrability is limited. We evaluate the implementation in both simulation and real robot experiments, assess the performance both qualitatively based on the trajectories generated by the controllers, and quantitatively based on the the robot alignment and heading errors as described in V-A.

1) Qualitative Comparisons of Generated Trajectories

Fig. 10 shows the trajectories generated by each MPC-based approach, in two selected scenarios (S_5 and S_6) from nine different experiment scenarios (see Table. 3 in Appendix E). We can see that in general, VMPC and BMPC approaches generate fairly similar trajectory characteristics, while the trajectories generated by SMPC have similar characteristics with the trajectories generated by HiDO-MPC method. This phenomenon can be explained based on the different objective functions implementation in these four MPC methods (see IV-C and Appendix D). VMPC and BMPC use a single objective function, in which the task (i.e. casualty approach procedure) is interpreted as a single objective task. On the other hand, in the SMPC and HiDO-MPC methods the casualty approach task is decomposed into several sub-objectives, so that these methods use multiple-objective functions corresponding to sequences, in SMPC, or stages, in HiDO-MPC.

In particular, experimental scenarios S_5 and S_6 highlight a key benefit of the decomposed-objective methods. As we can see, single-objective-based controllers generate oscillatory trajectories, because the controllers attempt to balance heading error and position error at the same time. More specifically, in the experimental scenarios S_6, VMPC controller fails to achieve the desired target pose. Multiple-objective approaches, SMPC and HiDO-MPC, have minimal oscillations, as the controllers are able to first minimise positional error then correct for heading error by making a single reversing move.

2) Quantitative Comparisons using Performance Metrics

In order to quantitatively compare the performance of each controller, in the real robot experiments and simulation, we use distance to heading line [cm] and heading error [°] as performance metrics.

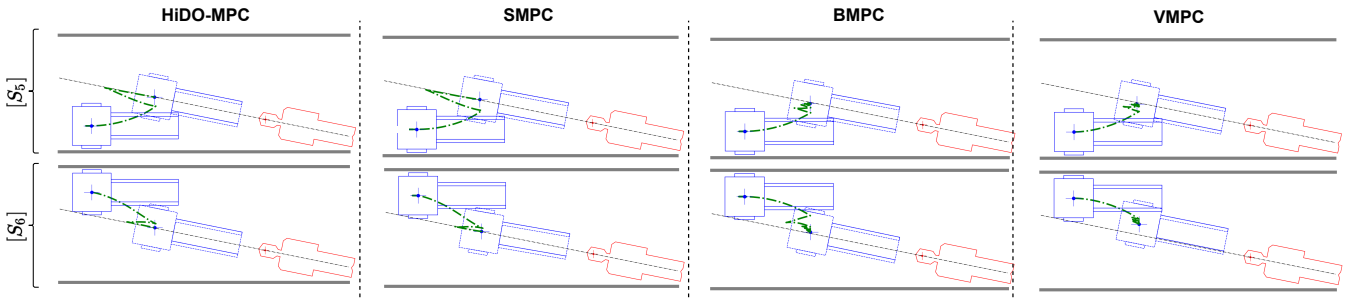


FIGURE 10. **Comparison of the trajectories generated in simulation by four different MPC methods (HiDO-MPC, SMPC, BMPC, and VMPC), in two selected experimental scenarios, S_5 and S_6 (see the scenario setup in Table 3 in Appendix E).**

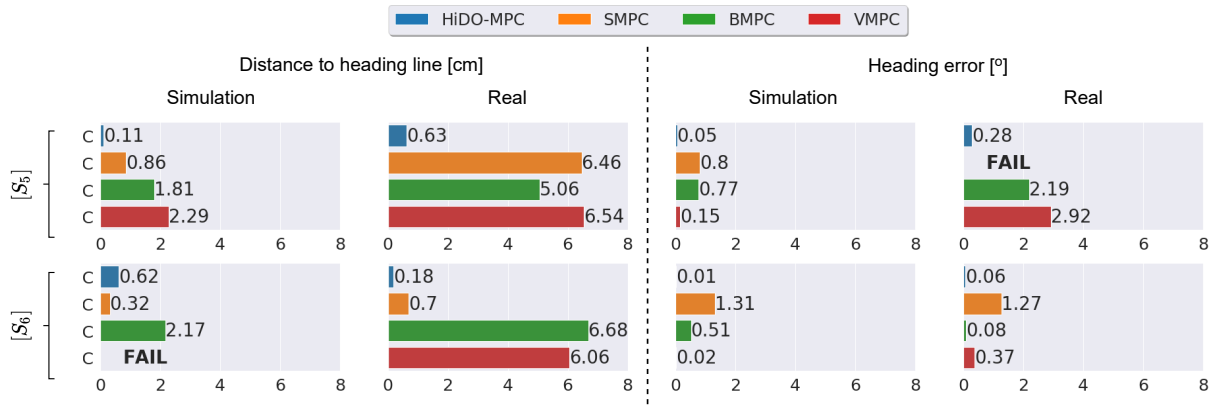


FIGURE 11. **Experimental results using two main quantitative metrics, *distance_to_heading_line* error [cm] and heading error [°] showing both simulation and real robot experiments, in two selected experimental scenarios, S_5 and S_6 (see the scenario setup in Table 3 in Appendix E).**

Fig. 11 presents the values of these metrics for each of the control method, in simulation and real robot, for selected experimental scenarios S_5 and S_6 (see the scenario setup in Table 3 in Appendix E. In general, the proposed HiDO-MPC approach shows superior performance compared to the other approaches based on the used metrics, in both simulation and real robot experiments. Results show that approaches with multiple decomposed objective functions, SMPC and HiDO-MPC, outperform approaches with a single objective function, VMPC and BMPC (especially with respect to the alignment metric), which is in line with our findings from the qualitative analysis. Scenarios S_5 and S_6, where the robot's initial pose is close to the wall highlight the robot manoeuvrability limitation, thus reaching the desired heading requires the robot to turn towards the wall. The approaches utilise single objective function like BMPC and VMPC have difficulties finding an optimal trajectory in such cases, because they are incentivised to fulfil all sub-objectives simultaneously. In contrast, SMPC and HiDO-MPC approaches manage to produce more consistent performances across all different scenarios.

In the simulation results SMPC and HiDO-MPC generally produce better performances compared to BMPC and VMPC. However, this is not the case for real world scenarios. While HiDO-MPC results significantly better performance compared to the other methods, in real scenarios S_5 and S_6 SMPC results the worst performance. SMPC fails to maintain the heading error within the safety limits in scenarios S_5. This is likely because its pure sequential execution (see Fig. 14 in Appendix D) ignores previous objectives. Minimising the heading error is the second objective, and is free to vary as the third objective, distance-to-point, is minimised. In simulation this is not a problem, as the previous objectives mean that the distance-to-point objective is achieved by driving forward in the reference frame of the robot. In reality, measurement error and control disturbances mean that driving forwards does not necessarily maintain heading. This finding further supports the conclusions reached from the obstacle density experiments with presence of noise and disturbance discussed in Section VI-B.

3) Controller Output Comparisons

Fig. 12 shows the control outputs (i.e. corresponding linear velocity $[v]$ and angular rate $[\omega]$ profiles) generated by all the evaluated controllers, and the corresponding actual final robot poses from the real-time physical robot experiment scenario S_5. The figure clearly emphasises the differences between the generated linear and angular velocity profiles in simulation and reality. The oscillations that occur are a result of noise and control disturbances that are inherent to physical systems. Nevertheless, we can see that HiDO-MPC produces minimal oscillations compared to other controllers.

Single objective controllers produce large oscillations in both simulation and reality, as they attempt to balance all objectives at any one time. Decomposed objective controllers,

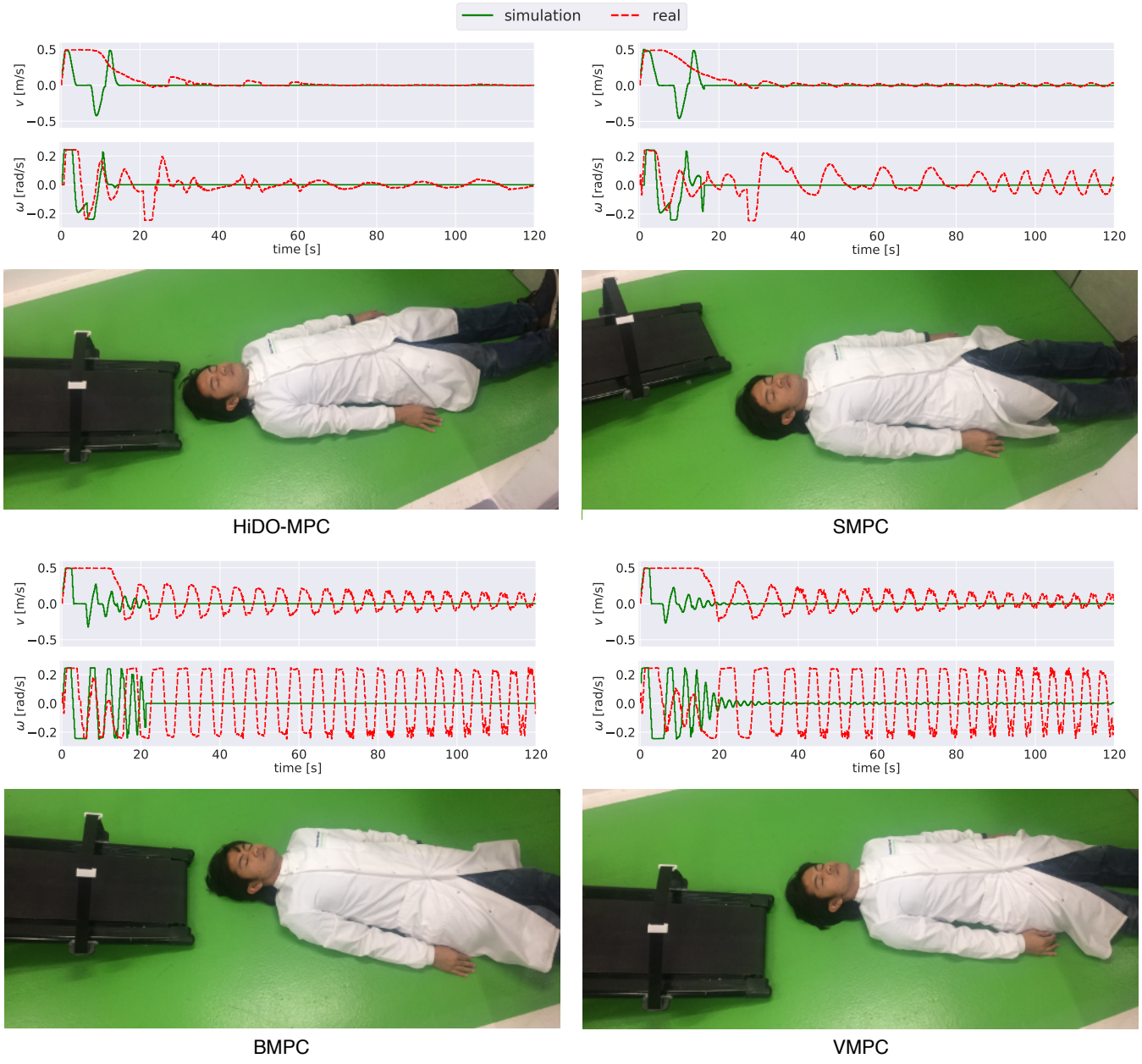


FIGURE 12. **Comparison of the controller output profiles (top), and the final robot pose (bottom) in the real robot experiment scenario [S_5] for each of the methods considered.**

SMPC and HiDO-MPC, are able to suppress oscillation significantly. However, the real robot angular velocity control output profile of SMPC exhibits an increase of the oscillation frequency. This occurs as the controller changes from minimising the heading error objective, to minimising the distance-to-point objective, and is unable to improve previous stages. On the contrary, HiDO-MPC is able to maintain the previous objective while minimising the next one. This allows HiDO-MPC to be more robust to the measurement noise and control disturbances present in a real-world scenario, to the point where it is able to achieve the desired pose within the safety limits in only 30 seconds.

Complete corridor experiment results, including generated trajectory from simulation and the quantitative metric results from corridor experiment scenarios [$S_1 - S_9$] can be found in Appendix F.

E. END-TO-END CASUALTY EXTRACTION SCENARIO

In addition to the experimental evaluations, we test our proposed HiDO-MPC controller within a real-life full end-to-end autonomous casualty extraction procedure using ResQbot. Fig. 13 presents the snapshots of ResQbot performing casualty extraction via the loco-manipulation procedure, by picking up the casualty — which is a real person — from

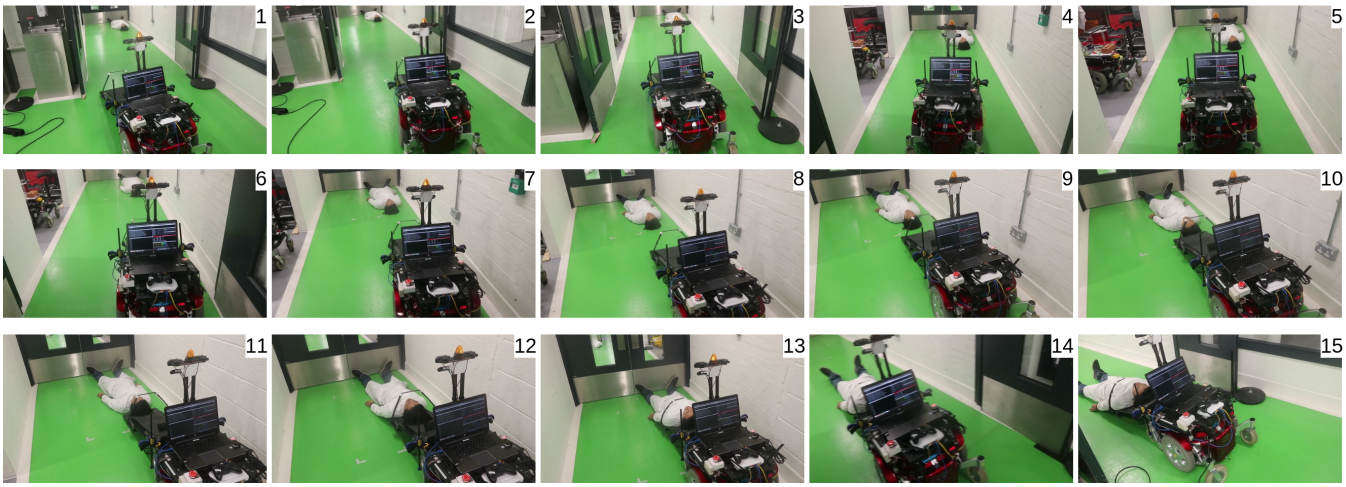


FIGURE 13. **Sequential snapshots of the end-to-end casualty extraction experiment executing a complete casualty approach and loco-manipulation procedure.**

the floor, onto the mobile robot and then transporting it to safety.

This experiment demonstrates the feasibility of integrating the HiDO-MPC controller within the full casualty extraction procedure described in Fig. 1. We find that the ResQbot is able to successfully perform a complete casualty extraction procedure safely, without human intervention (i.e. autonomously).

Additional experimental results, including videos, can be found on the project website[3].

VII. CONCLUSION

In this paper, we propose a hierarchical decomposed-objective model predictive control (HiDO-MPC) method to control a mobile rescue robot safely approaching a casualty, as a part of a casualty extraction procedure. The HiDO-MPC approach is inspired by the hierarchical-sequential nature of the casualty extraction procedure as performed by experienced teleoperators. Our hypothesis is that the proposed approach fulfils the controller design requirements for achieving the casualty approaching task, including task accomplishment with safety metrics (as the main priority), and collision free operation.

A series of experiments has been systematically conducted to evaluate the hypothesis, and to compare the performance of the proposed method to other methods (adapted from the state of the art), both in simulation and on the real robot experiments. Inspired by the current literature, we use SMPC, BMPC, and VMPC as comparison methods. ResQbot, a proof-of-concept mobile rescue robot that we have previously developed, is used for the real-robot experiments.

The results of the experiments with several different obstacle densities demonstrate that HiDO-MPC — which uses multiple decomposed objective functions in a hierarchical execution — significantly outperforms the methods, such as BMPC and VMPC — which use a single objective function — in controlling ResQbot to approach the target casualty, while satisfying the collision free constraint. Moreover, the experimental results which incorporate state estimation error and control perturbations highlight the fact that HiDO-MPC is robust to the presence of such modelling errors, while the SMPC performance is significantly affected by these. This robustness of HiDO-MPC to modelling errors is very important when implementing the controller on a real physical system, in which disturbances and measurement noise is inevitable. Additionally, regarding the time effectiveness, the experimental results show that by using HiDO-MPC, ResQbot reaches the desired target pose faster than with using other methods.

A case study of controlling ResQbot approaching a casualty in a narrow corridor, has been conducted to evaluate the performance of the proposed HiDO-MPC in both simulation and real-system experiments. HiDO-MPC outperforms other methods in both simulation and real experiments. More specifically, the real-time physical robot experiments highlight the fact that the HiDO-MPC method demonstrates robust performance while dealing with uncertainty occurring in the real scenarios, and shows potential for a realistic rescue scenario.

Further work on developing a realistic simulation with physics engine environments could be leveraged to improve and evaluate the robustness of the controller, in a wider-range of real-world scenarios. A significant amount of environment scenarios could be developed in simulation — which could considerably mimic the actual environment — and further used to tune, evaluate and improve the proposed approach prior to deployment in real-world missions. Another potential extension work from this study is to explore the potential implementation of the proposed HiDO-MPC (i.e. Algorithm 1) for a wider practical applications, especially for a high-dimensional task that can be decomposed into subsequent tasks (e.g. locomotion and manipulation tasks).

[3]https://sites.google.com/view/hido-mpc-resqbot/experiments

ACKNOWLEDGEMENTS

The authors would like to thank David Yung who has spent a large amount of time and effort for helping them during the experiments and James Paul Foster for the initial discussion leading to this work. The experiment video (simulation and real robot), additional material, as well as the code used for the experiments, will be publicly available on the project website: https://sites.google.com/view/hido-mpc-resqbot upon publication.

.

APPENDIX A DETAILED DERIVATION OF DISTANCE-TO-LINE OBJECTIVE

This objective function is formulated to achieve the robot alignment subtask.

Given a finite heading line segment H (an imaginary line along the casualty orientation, see Fig. 3), where the line segment is defined through two points, starting point $\mathbf{p}^s = (x^s, y^s)$ and ending point $\mathbf{p}^e = (x^e, y^e)$, and the robot position point $\mathbf{p}^r = (x^r, y^r)$, let $d(\mathbf{p}^r, H)$ denote the minimum distance from $\mathbf{p}^r$ to the line segment H. This is the shortest distance separating $\mathbf{p}^r$ and H. Since H is a finite line segment, then $d(\mathbf{p}^r, H)$ is the orthogonal distance between point $\mathbf{p}^r$ and the nearest point along line H.

Let $\mathbf{p}$ denote the point belonging to the line through the line segment H defined as:

$$\mathbf{p} = \mathbf{p}^s + u(\mathbf{p}^e - \mathbf{p}^s). \tag{14}$$

The point $\mathbf{p}$ is the projection of the point $\mathbf{p}^r$ onto the line segment H if the dot product between vector $(\mathbf{p}^r - \mathbf{p})$ and vector $(\mathbf{p}^e - \mathbf{p}^s)$ is equal to zero.

$$(\mathbf{p}^r - \mathbf{p})\cdot(\mathbf{p}^e - \mathbf{p}^s) = 0. \tag{15}$$

By substituting Eq. 14 to Eq. 15, the value u can be calculated as:

$$u = \frac{(x^r - x^s)(x^e - s^s) + (y^r - y^s)(y^e - y^s)}{\|\mathbf{p}^e - \mathbf{p}^s\|^2}. \tag{16}$$

Hence, the nearest point coordinate $\mathbf{p}(x, y)$ can be determined as:

$$\begin{aligned} x &= x^s + u(x^e - x^s) \\ y &= y^s + u(y^e - y^s) \end{aligned} \tag{17}$$

Since the line segment H is constrained between point $\mathbf{p}^s$ and $\mathbf{p}^e$, the scaling parameter u is also constrained between 0 and 1. The distance-to-line segment objective function can then be formulated as:

$$\begin{aligned} F_1(\mathbf{x}^r, \mathbf{x}^c) \quad &= d(\mathbf{p}^r, H) = \|\mathbf{p}^r - \mathbf{p}\|^2 \\ \text{s.t.} \\ \mathbf{p} &= \mathbf{p}^s + u(\mathbf{p}^e - \mathbf{p}^s) \\ u &= \begin{cases} u, & 0 \le u \le 1 \\ 0, & u < 0 \\ 1, & u > 1 \end{cases} \end{aligned} \tag{18}$$

APPENDIX B COLLISION FUNCTIONS

We denote the state of each circle as $\mathbf{c} = [x_{cr}, y_{cr}, r_{cr}]$, where (x_{cr}, y_{cr}) is the position of the circle and r_{cr} is the radius of the circle. The collision function between two circles is then defined as:

$$\begin{aligned} [col(\mathbf{c}^1, \mathbf{c}^2)]_+ = [(r^a_{cr} + r^b_{cr})^2 - (x^a_{cr} - x^b_{cr})^2 \\ - (y^a_{cr} - y^b_{cr})^2]_+ \end{aligned} \tag{19}$$

We then model the collision geometry of ResQbot as n_{c^r} number of circles, so that:

$$\mathbf{c}^r(\mathbf{x}^r, r^r_{cr}, n_{c^c}) = [x^r_{cr_h}, y^r_{cr_h}, r^r_{cr_h}], \quad \forall\, h \in [0, n_{c^c}] \tag{20}$$

Similarly, the collision geometry of the casualty can be expressed as:

$$\mathbf{c}^c(\mathbf{x}^c, r^c_{cr}, n_{c^r}) = [x^c_{cr_i}, y^c_{cr_i}, r^c_{cr_i}], \quad \forall\, i \in [0, n_{c^c}] \tag{21}$$

Let $\mathbf{p}^{obs}$ denote the vectors of size n^{obs}_{cr} that describe the obstacle position and $\mathbf{r}^{obs}_{cr}$ that describe the obstacle radius. The collision geometry of the obstacles can then be written as:

$$\begin{aligned} \mathbf{c}^{obs}(\mathbf{p}^{obs}, \mathbf{r}^{obs}_{cr}, n^{obs}_{cr}) = [x^{obs}_{cr_j}, y^{obs}_{cr_j}, r^{obs}_{cr_j}], \\ \forall\, j \in [0, n_{c^{obs}}] \end{aligned} \tag{22}$$

Thus, the vector of obstacle collision constraints and the casualty collision constraints can be formulated as:

$$\begin{aligned} [\mathbf{col}_{obs}(\mathbf{c}^r, \mathbf{c}^{obs})]_+ &= \bar{0} \\ [\mathbf{col}_{cas}(\mathbf{c}^r, \mathbf{c}^c)]_+ &= \bar{0} \end{aligned} \tag{23}$$

Then we can include this collision constraint formulation as equality expressions such that $[col]_+ = 0$ means that the collision constraint is satisfied.

APPENDIX C TRANSFORMING HIDO-MPC INTO A NLP PROBLEM

To transform our finite horizon OCP problem into a NLP problem formulation, we implement the direct single-shooting technique. We define the NLP decision variable $\mathbf{u}$ along the finite horizon N as:

$$\mathbf{u} = [\mathbf{u}_{|0}, \cdots, \mathbf{u}_{|N-1}] \tag{24}$$

The robot state trajectory $\mathbf{X}^r$ along the horizon N can be explicitly expressed as a recursive function of the control trajectory using the robot's dynamic function.

$$\begin{aligned} \mathbf{X}^r &= [\mathbf{x}^r_{|0}, \cdots, \mathbf{x}^r_{|N-1}] \\ &= \mathbf{F}(\mathbf{u}, \mathbf{x}_{|0}) \end{aligned} \tag{25}$$

where:

$$\mathbf{x}_0 = \mathbf{x}^r_{|0}$$

The corresponding objective function w.r.t. current stage, $\mathbf{J}^s_N$ is a combined function of the three objective function components, F_1, F_2, F_3, and the control decision variable $\mathbf{u}$ can be expressed as:

$$J^s_N = \sum_{k=0}^{N-1} J^s(F_1(\mathbf{x}^R_{|k}, \mathbf{x}^c), F_2(\mathbf{x}^r_{|k}, \mathbf{x}^c), F_3(\mathbf{x}^r_{|k}, \mathbf{x}^c), \mathbf{u}_{|k}), \tag{26}$$

TABLE 1. The optimal cost function weight values of each evaluated MPC controller obtained via Bayesian Optimisation.

MPC Type	Weights								
	1	2	3	4	5	6	7	8	9
VMPC — W^V	99.91	1.04	-	-	-	-	-	-	-
BMPC — W^B	97.97	95.66	1.63	97.29	-	-	-	-	-
SMPC — W^S	7.75	3.23	96.51	48.32	99.69	10.71	-	-	-
HiDO-MPC — W^H	36.57	1.13	68.02	60.09	96.78	98.01	93.04	95.28	71.16

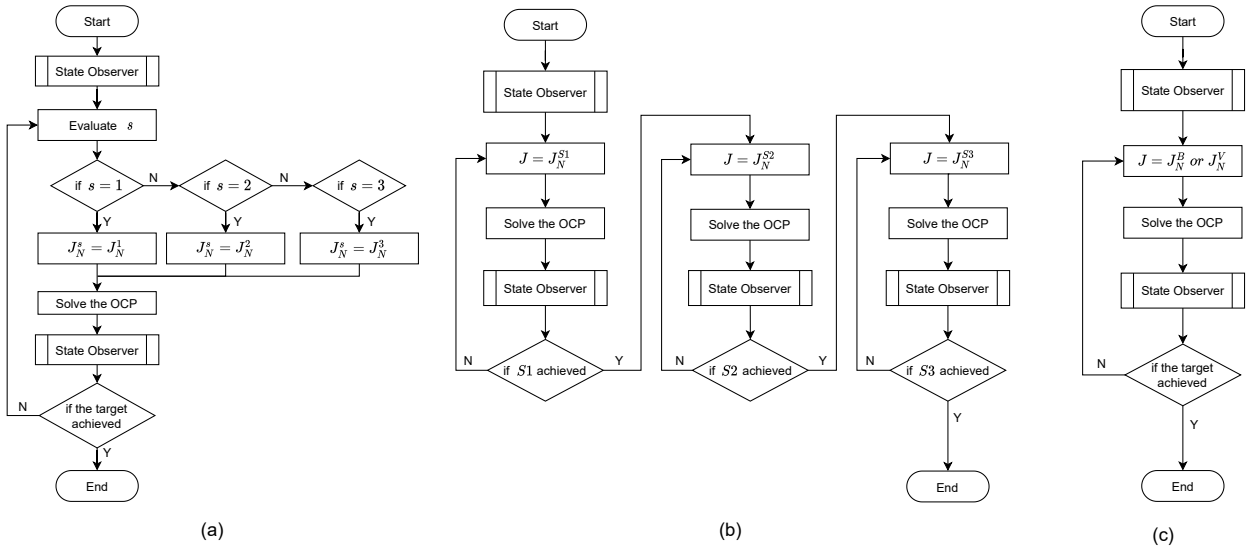


FIGURE 14. Flowcharts illustrate the implementation of different MPC controllers: (a) HiDO-MPC, (b) SMPC, (c) BMPC and VMPC.

where:

$$F_1(\mathbf{x}^r_{|k}, \mathbf{x}^c) = d(\mathbf{p}^r(\mathbf{x}^r_{|k}), H(\mathbf{x}^c))$$
$$F_2(\mathbf{x}^r_{|k}, \mathbf{x}^c) = \Delta(\phi^r(\mathbf{x}^r_{|k}), \phi^t(\mathbf{x}^c))$$
$$F_3(\mathbf{x}^r_{|k}, \mathbf{x}^c) = \left\| \mathbf{p}^r(\mathbf{x}^r_{|k}) - \mathbf{p}^t(\mathbf{x}^c)) \right\|^2$$

The parameters H, ϕ^t, and $\mathbf{x}^t$ can be derived explicitly as a function of the casualty states, $\mathbf{x}^c$. Thus, the function can be simplified and written as:

$$\mathbf{J}^s_N = \sum_{k=0}^{N-1} J^s(\mathbf{u}_{|k}, \mathbf{x}^r_{|k}, \mathbf{x}^c). \tag{27}$$

The equality constraints are formulated from the collision avoidance constraints (see subsection IV-D), and can be re-expressed as:

$$\mathcal{C}(\mathbf{c}^r(\mathbf{x}^r, r^r_{cr}, n_{c^c}), \mathbf{c}^c(\mathbf{x}^c, r^c_{cr}, n_{c^r}), \mathbf{c}^{obs}(\mathbf{p}^{obs}, \mathbf{r}^{obs}_{cr}, n^{obs}_{cr})) = 0 \tag{28}$$

For all the robot state trajectory prediction $\mathbf{x}^r$ along the finite horizon N, these constraints can be written as:

$$\mathbf{C}_N(\mathbf{u}, \mathbf{x}^r_0, \mathbf{r}^r_{cr}, n_{c^r}, \mathbf{x}^c, \mathbf{r}^c_{cr}, n_{c^c}, \mathbf{p}^{obs}, \mathbf{r}^{obs}_{cr}, n^{obs}_{cr}, k) = 0, \quad \forall\, k \in [0, N-1] \tag{29}$$

Thus, from the objective function, Eq. 27, and the constraint equality function, Eq. 29, the parametric NLP optimisation problem can be written as:

$$\min_{\mathbf{u}} : \mathbf{J}^s_N(\mathbf{u}, \xi) \quad \text{subject to: } \mathbf{C}_N(\mathbf{u}, \xi) = 0, \tag{30}$$

in which ξ is a known parameter vector:

$$\xi = [\mathbf{x}^r_0, \mathbf{r}^r_{cr}, n_{c^r}, \mathbf{x}^c, \mathbf{r}^c_{cr}, n_{c^c}, \mathbf{p}^{obs}, \mathbf{r}^{obs}_{cr}, n^{obs}_{cr}, k] \tag{31}$$

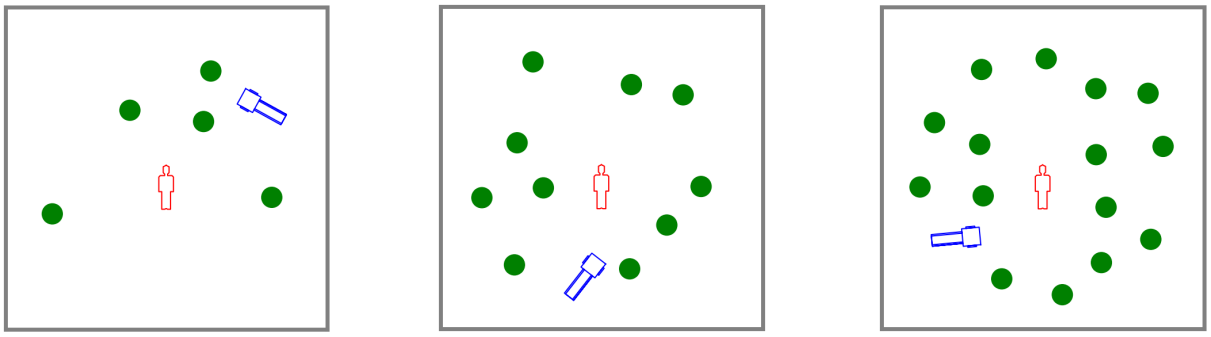
APPENDIX D IMPLEMENTATION COST FUNCTIONS IN COMPARED METHODS (ADAPTED FROM STATE OF THE ART)

SMPC: This approach uses the same cost function components, F_1, F_2, F_3, as HiDO-MPC. However, in this method, the three objective function components are independently executed in a sequential order that corresponds to the subsequent-tasks (see IV-A). These sequential objective functions can be written as:

- **Objective Sequence 1**

$$J^{S1}_N = \sum_{k=0}^{N-1} W^S_1 F_1 + W^S_2 \left\| \mathbf{u}(\cdot) - \mathbf{u}^t(\cdot) \right\|^2_{\mathbf{R}} \tag{32}$$

- **Objective Sequence 2**



(a) Low Density (5 random obstacles setup)

(b) Medium Density (10 random obstacles setup)

(c) High Density (15 random obstacles setup)

FIGURE 15. **Samples of generated environments with three different obstacle density levels: (a) low density, (b) medium density, and (c) high density.**

TABLE 2. **Different combinations of the state estimation error and control perturbations applied in the simulation experiments. The values presented in the table are chosen to represent the sensor measurement errors and the controller imperfections in real robot system.**

Noise Combination ID	State Estimation Error ($\mathcal{E}$)						Control Perturbation Gain ($\mathcal{G}$)			
	$\mathcal{E}_x\ [m]$		$\mathcal{E}_y\ [m]$		$\mathcal{E}_\phi\ [rad]$		$\mathcal{G}_v$		$\mathcal{G}\omega$	
	μ_x	σ_x	μ_y	σ_y	μ_ϕ	σ_ϕ	μ_v	σ_v	μ_ω	σ_ω
1	0.0	0.01	0.0	0.01	-	-	-	-	-	-
2	0.0	0.05	0.0	0.05	-	-	-	-	-	-
3	0.0	0.1	0.0	0.1	-	-	-	-	-	-
4	-	-	-	-	0.0	0.01	-	-	-	-
5	-	-	-	-	0.0	0.05	-	-	-	-
6	-	-	-	-	0.0	0.1	-	-	-	-
7	-	-	-	-	-	-	0.0	0.25	-	-
8	-	-	-	-	-	-	0.25	0.25	-	-
9	-	-	-	-	-	-	-0.25	0.25	-	-
10	-	-	-	-	-	-	-	-	0.0	0.25
11	-	-	-	-	-	-	-	-	0.25	0.25
12	-	-	-	-	-	-	-	-	-0.25	0.25
13	0.0	0.1	0.0	0.1	0.0	0.1	-	-	-	-
14	-	-	-	-	-	-	0.25	0.25	0.25	0.25
15	0.0	0.1	0.0	0.1	0.0	0.1	0.25	0.25	0.25	0.25

$$J_N^{S2} = \sum_{k=0}^{N-1} W_3^S F_2 + W_4^S \left\|\mathbf{u}(\cdot) - \mathbf{u}^t(\cdot)\right\|_{\mathbf{R}}^2 \tag{33}$$

- **Objective Sequence 3**

$$J_N^{S3} = \sum_{k=0}^{N-1} W_5^S F_3 + W_6^S \left\|\mathbf{u}(\cdot) - \mathbf{u}^t(\cdot)\right\|_{\mathbf{R}}^2 \tag{34}$$

BMPC: In contrast to HiDO-MPC and SMPC, BMPC approach uses a single weighted combination function of all three cost function components, F_1, F_2, F_3, to generate a single robot motion behaviour. This BMPC objective function can be written as:

$$J_N^B = \sum_{k=0}^{N-1} W_1^B F_1 + W_2^B F_2 + W_3^B F_3 + W_4^B \left\|\mathbf{u}(\cdot) - \mathbf{u}^t(\cdot)\right\|_{\mathbf{R}}^2 \tag{35}$$

VMPC: This method is a baseline of generic MPC implementation for controlling a mobile robot. This approach uses a simple objective function that can be written as:

$$J_N^V = \sum_{k=0}^{N-1} W_1^V \left\|\mathbf{x}(\cdot) - \mathbf{x}^t(\cdot)\right\|_{\mathbf{Q}}^2 + W_2^V \left\|\mathbf{u}(\cdot) - \mathbf{u}^t(\cdot)\right\|_{\mathbf{R}}^2 \tag{36}$$

The implementation of these three controllers in comparison to the proposed HiDO-MPC controller in the optimisation framework is illustrated through the flowcharts in Fig. 14. Flowchart (a) demonstrates the implementation of the hierarchical decomposed-objective approach of the HiDO-MPC. In contrast, flowchart (b) shows the implementation of the sequential execution of the three subsequent tasks of the SMPC. Lastly, flowchart (c) illustrates the generic MPC implementation used for the BMPC and VMPC methods.

APPENDIX E DETAILED EXPERIMENTAL SETUP

A. OBSTACLES DENSITY EXPERIMENTS

In this experimental setup, we investigate the performance of each controller when executing the task in environments with different obstacle density distributions.

Details of the experimental setup are described as follows:

- In total, we generate 3 [densities] $\times$ 600 [samples] environment setups.
- Each environment consists of a casualty with fixed pose in the centre of the map, a ResQbot with a random initial pose, and a set of obstacles which are uniformly distributed around the casualty in a circular pattern.
- We evaluate three different obstacle density levels: low, medium and high, having 5, 10, and 15 objects, respectively.
- The collision geometry of each obstacle is modelled as a circle with a radius of 35 cm.
- Obstacles are fixed and the minimum distance between the obstacles is 1 m, to ensure execution feasibility.
- The obstacles are uniformly distributed around the casualty within a 4.5 m radius from the centre (i.e. the casualty position).
- The robot's initial pose is randomly generated with the following constraints:
  - The robot's initial position is randomly sampled on a circle of fixed radius ($r = 4.5$ m) centred on the casualty (i.e. centre of the map).
  - The robot's initial orientation ϕ_{init} is uniformly sampled from $[-\pi, \pi]$.

Figure 15 shows examples of generated environments with three different obstacle density levels.

B. STATE ESTIMATION ERROR AND CONTROL PERTURBATION EXPERIMENTS

MPC controllers generate optimal control signals based on the current robot pose provided by the state estimation module, and send these control signals to the robot via the low-level controller module (i.e. linear and angular velocity controllers). In reality, the state estimation and low-level controller modules are imperfect. Inaccuracies present in the state estimation are inevitable, and are mostly due to the sensor measurement errors. Controller imperfections are usually caused by inaccuracies in the control signal transmission or environmental effects inherent to the real physical system.

We take into account these sources of uncertainties occurring in the physical system, by modelling them within the simulation experiments. We then evaluate the controller performances in these situations based on the failure ratio metric.

State Estimation Error: We model the state estimation error to simulate the uncertainty of the state estimation module in the real robot perception system. This is achieved by adding noise to the corresponding elements of the robot's state vector, $\mathbf{x} = [x, y, \phi]^\intercal$. We denote the state estimation error vector as $\mathcal{E}$, where each component $\mathcal{E}_i$ is modelled as a stochastic variable with a normal distribution $\mathcal{E}_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, where i is the index the vector elements. We design several experiments by selectively applying $\mathcal{E}_i$ to corresponding elements of the state vector $\mathbf{x}_i$, in order to analyse the contribution of different noise types. The state estimation with uncertainty is calculated as:

$$\hat{\mathbf{x}} \sim \mathbf{x} + \mathcal{E} \tag{37}$$

where:
$\hat{\mathbf{x}}$: state estimation vector with uncertainty,
$\mathbf{x}$: actual value of the state estimation vector,
$\mathcal{E}$: state estimation error.

Control Perturbation: In a differential drive mobile robot system such as ResQbot, imperfections in the dynamics model could occur due to non-uniform terrain friction, wheel slippage or transmission imperfections. In this experiment, we simulate the controller imperfections as a perturbation of the control signal proportional to the calculated optimal control output. This perturbation corresponds to each element of the control decision vector $\mathbf{u} = [v, \omega]^\intercal$. We denote the perturbation gain as $\mathcal{G}$, where each component of the vector $\mathcal{G}_i$ is modelled as a stochastic variable with a normal distribution $\mathcal{G}_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, and represents the gain applied to each corresponding element of the control decision vector $\mathbf{u}_i$. The perturbed control decision vector is calculated as:

$$\mathbf{u}_{exe} \sim (1 + \mathcal{G})\mathbf{u}_{cal} \tag{38}$$

where:
$\mathbf{u}_{exe}$: perturbed control signal,
$\mathbf{u}_{cal}$: calculated optimal control signal,
$\mathcal{G}$: control perturbation gain,

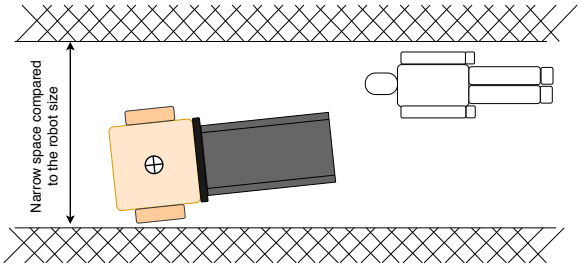


FIGURE 16. The illustration of the narrow corridor experiments conducted in this study. In these experiments, we implement the MPC controllers for approaching casualties in an environment in which the robot's manoeuvrability is highly restricted.

TABLE 3. Different combinations of initial robot pose and the target casualty pose used in the 1.8 m-wide corridor experiments (see Fig. 16).

Scenario	Initial Robot Pose			Target Casualty Pose		
	$\mathbf{x}^r$	$\mathbf{y}^r$	ϕ^r	$\mathbf{x}^t$	$\mathbf{y}^t$	ϕ^t
S_1	0.0	0.0	0.0	3.2	0.0	0.0
S_2	0.5	-0.5	0.0	3.2	0.0	0.0
S_3	0.5	0.5	0.0	3.2	0.0	0.0
S_4	0.0	0.0	0.0	3.2	-0.4	-11.5
S_5	0.5	-0.5	0.0	3.2	-0.4	-11.5
S_6	0.5	0.5	0.0	3.2	-0.4	-11.5
S_7	0.0	0.0	0.0	3.2	0.4	11.5
S_8	0.5	-0.5	0.0	3.2	0.4	11.5
S_9	0.5	0.5	0.0	3.2	0.4	11.5
Top wall	$y = 0.9$					
Bottom wall	$y = -0.9$					

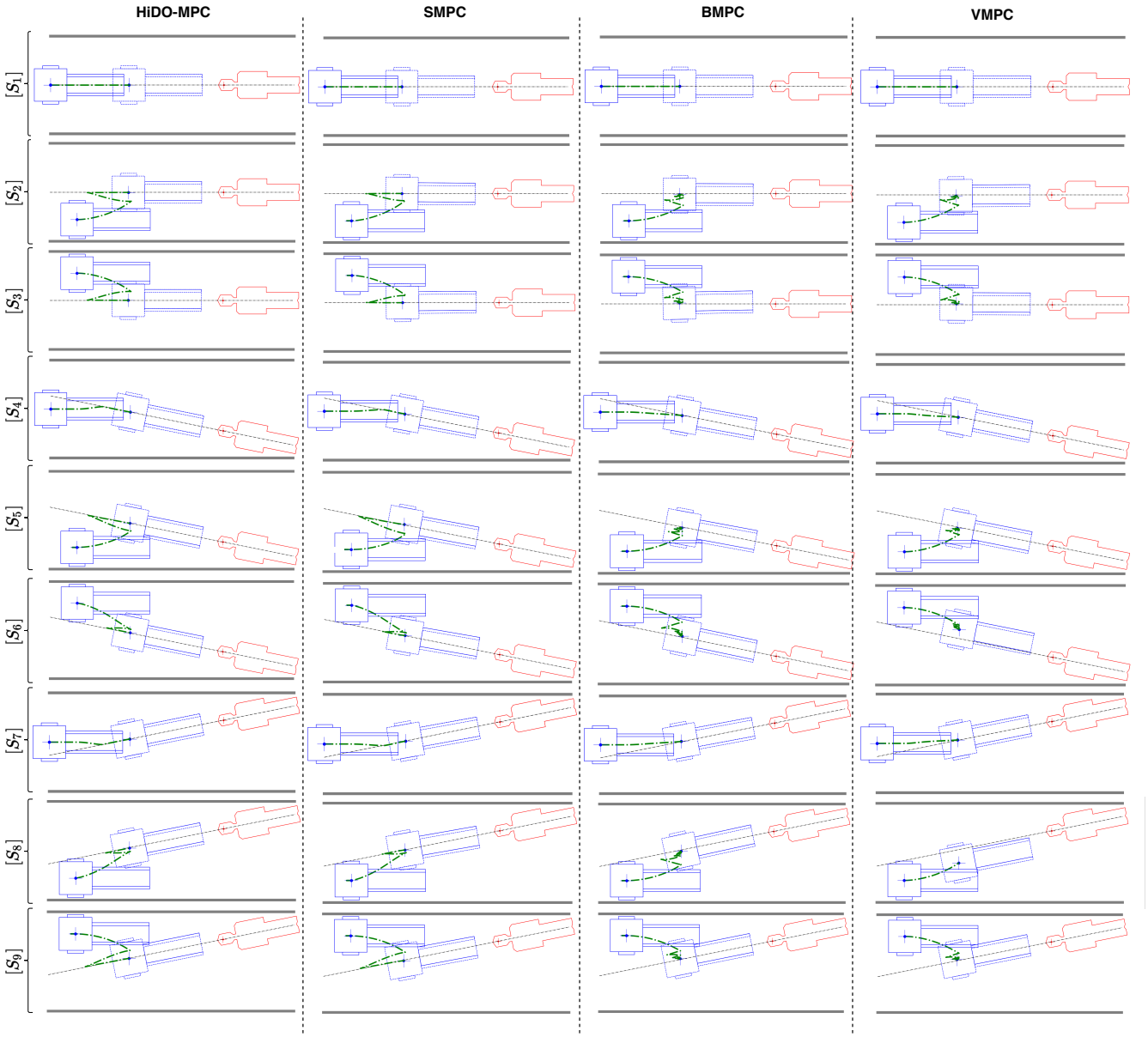


FIGURE 17. A complete trajectory comparison generated in simulation by four different MPC methods (HiDO-MPC, SMPC, BMPC, and VMPC), in all corridor experiment scenarios [$S_1 - S_9$].

To examine the contributions of the above-mentioned uncertainty factors, we perform the following steps:

- We use the same environment setups as for the experiments described in the Appendix E-A, to conduct the uncertainty evaluation scenarios.
- We evaluate 15 different experimental scenarios, each using a different combinations of uncertainties applied to different elements of the state vector or the control signal, presented in Table 2.
- For each uncertainty combination, we conduct the same number of experimental sets, as in the obstacle density experiments (i.e. 600 experimental trials for each density, with random initial robot pose and random obstacle distribution).
- We then evaluate the performance of the HiDO-MPC in comparison to SMPC, BMPC, and VMPC, by calculating their failure ratios.

C. EXECUTION TIME COMPARISON

Execution time performance is calculated based on the required time for each evaluated controllers to navigate ResQbot from its starting pose to the target pose. This evaluation is subject to the main criteria, alignment and heading error thresholds (see Section V-B), meaning that the execution time is only considered when the target pose has been achieved successfully.

Since each initial configuration is random, the execution time for individual configurations cannot be compared di-

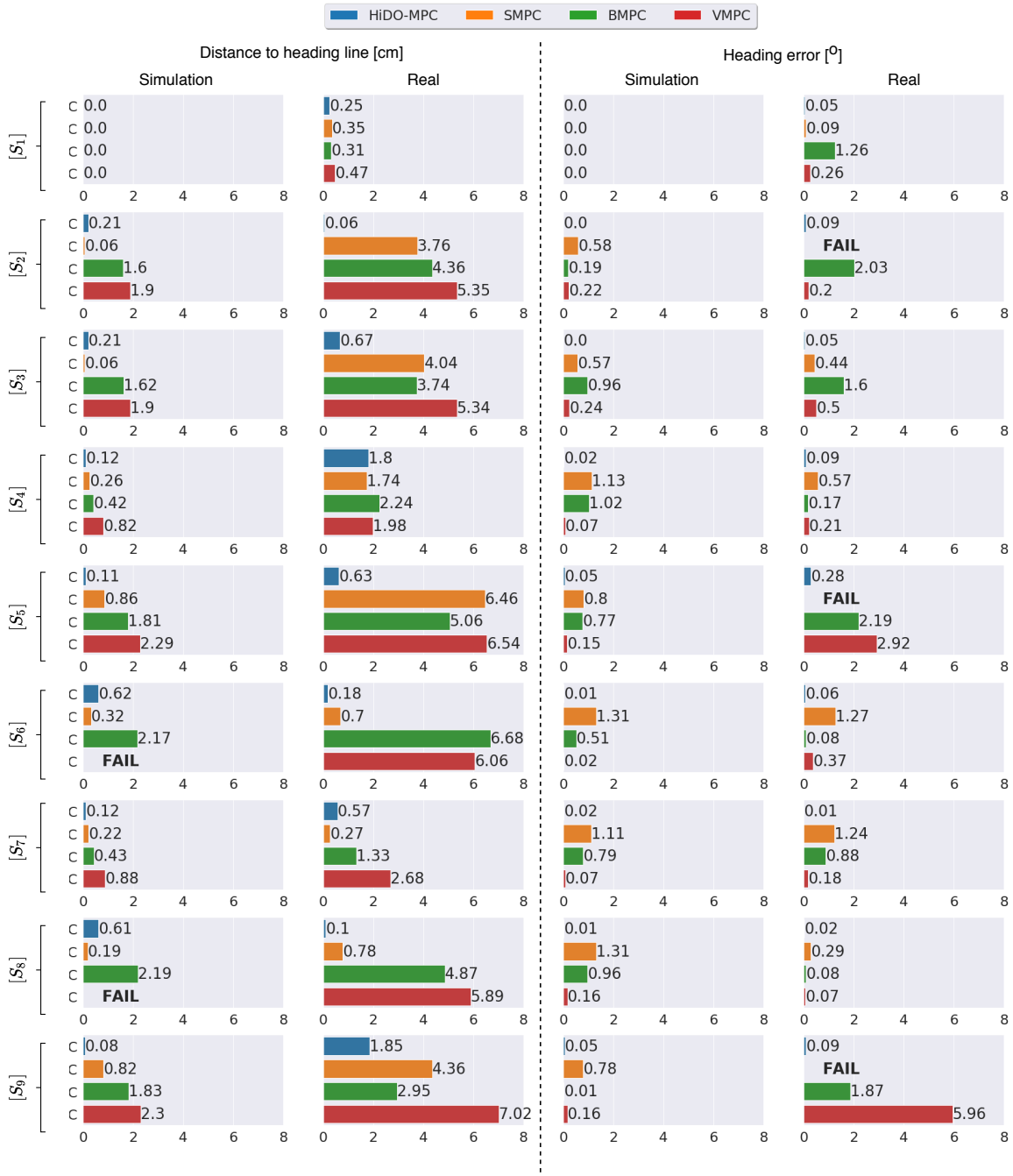


FIGURE 18. Complete quantitative experimental results showing both simulation and real robot experiments from all corridor experiment scenarios [$S_1 - S_9$].

rectly. Therefore, we perform the execution time comparison for other controllers, in each independent experiment, as the ratio with respect to the HIDO-MPC execution time as a baseline.

D. NARROW CORRIDOR EXPERIMENT

In this experimental scenario, we compare the performance of the MPC-based controllers when approaching a casualty in a narrow corridor environment. This experimental scenario is conducted to evaluate the performance of each controller when the robot's manoeuvrability is highly restricted (depicted in Fig. 16). To ensure the feasibility of the task, we restrict the scenario in two ways: (i) the casualty pose is restricted such that there is sufficient space for the robot to execute the approach safely, (ii) there are no other obstacles, except the corridor wall, that could block the robot from reaching the casualty.

In total, nine different narrow corridor scenarios are considered. Each scenario presents a combination of three initial robot poses and three casualty poses, within a narrow 1.8 m-

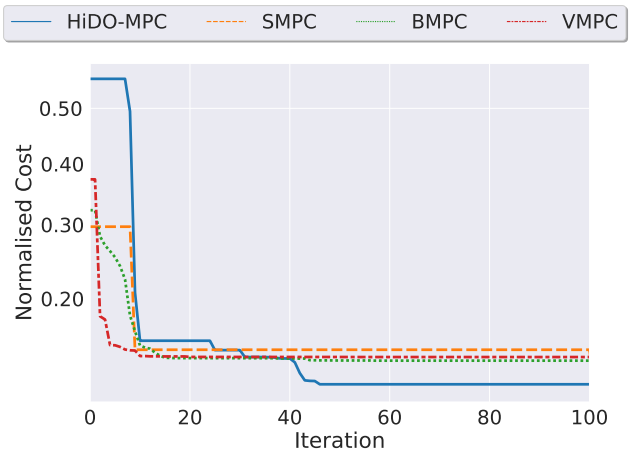


FIGURE 19. Normalised cost over Bayesian Optimisation iterations during the weight optimisation process (W^H, W^V, W^B and W^S) for HiDO-MPC, SMPC, BMPC, and VMPC.

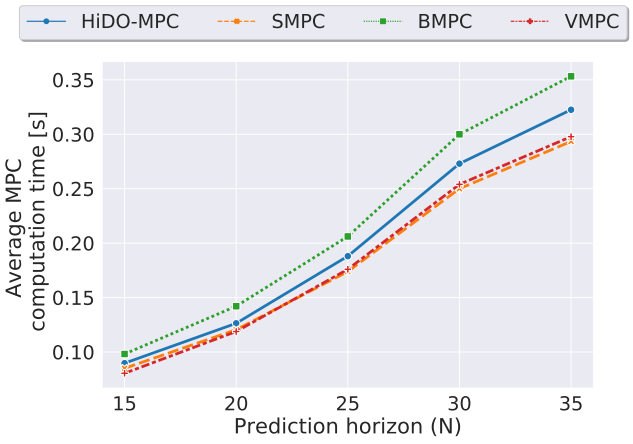


FIGURE 20. Comparison of average computation times for all four different MPC-based methods, using different prediction horizon lengths.

wide corridor. Table 3 summarises the narrow corridor scenarios.

APPENDIX F COMPLETE CORRIDOR EXPERIMENT RESULTS [$S_1 - S_9$]

Fig. 17 shows all the trajectories generated by each MPC approach (i.e. HiDO-MPC, SMPC, BMPC, VMPC), in all corridor experiment scenarios based on experimental setup described in Appendix E-D.

Fig. 18 presents the complete results of the quantitative evaluation of all corridor experiments scenarios, including simulation and real robot experiments.

APPENDIX G HYPER-PARAMETER TUNING RESULTS

A. TUNING MPC WEIGHTS VIA BAYESIAN OPTIMISATION

To find the optimal weighing coefficients for each MPC method (W^H, W^V, W^B and W^S), we use Bayesian Optimisation (BO) as presented in [62], as it is a sample-efficient global optimisation method.

We perform 100 iterations of BO in the corridor experiment setups (see Appendix E-D), using the default BO hyper-parameters presented in [63]. Each iteration consists of evaluating the current weight coefficient values for each MPC approach, on the training setup, in which we used the perturbed versions of test experiment scenarios. We define the cost for BO optimisation as the sum of the execution times over all 9 narrow corridor experiment scenarios. We then normalise the value over the maximum cost during the optimisation process. Fig. 19 shows the normalised cost function for each MPC approach over BO iterations, during the weight optimisation process; optimising W^H, W^S, W^B, and W^V for HiDO-MPC, SMPC, BMPC, and VMPC, respectively.

The optimisation results show that the optimisation processes converge for all evaluated MPC controllers. As we can see in Fig. 19, the convergence times of the weight optimisation process is closely correlated to the number of weight components in each cost function. The weight optimisation process for the VMPC cost function converges in less than 10 iterations, whereas HiDO-MPC cost function requires more than 40 iterations before it converges, and it reaches a lower overall value. Table 1 shows the optimal weight values for each of the MPC type cost functions, obtained via Bayesian Optimisation.

B. OPTIMAL MPC HORIZON

We compare the MPC computation times over five different prediction horizon lengths $N = \{15, 20, 25, 30, 35\}$, in order to confirm that the MPC frameworks can be implemented in real-time. The observation results show an almost linear correlation between the computation time and the number of prediction horizon, N, where VMPC and SMPC are slightly faster than HiDO-MPC due to their simpler cost function evaluations (see Fig. 20). Any of the MPC approaches with any of the prediction horizons have a computation time fast enough to be implemented in real-time.

REFERENCES

[1] A. C. Yoo, G. R. Gilbert, and T. J. Broderick, "Military robotic combat casualty extraction and care," in *Surgical Robotics: Systems Applications and Visions*. Springer U.S., 2011, pp. 13–32. [Online]. Available: https://doi.org/10.1007/978-1-4419-1126-1_2

[2] D. Theobald, "Mobile extraction-assist robot," U.S. Patent 7 719 222, May 18, 2010.

[3] B. Choi, W. Lee, G. Park, Y. Lee, J. Min, and S. Hong, "Development and control of a military rescue robot for casualty extraction task," *J. Field Robot.*, vol. 36, no. 4, pp. 656–676, 2019. [Online]. Available: https://doi.org/10.1002/rob.21843

[4] K. Osuka and S. Isayama, "Motion control of multi-linked stretcher robot ducks," in *Proceedings of SICE Annual Conference 2010*, Taipei, Taiwan, 2010, pp. 873–874.

[5] Y. Iwano, K. Osuka, and H. Amano, "Development of rescue support stretcher system," in *2010 IEEE International Symposium on Safety Security and Rescue Robotics*, Bremem, Germany, 2010, pp. 1–6.

[6] Y. Iwano, K. Osuka, and H. Amano, "Development of rescue support stretcher system with stair-climbing," in *2011 IEEE International Symposium on Safety Security and Rescue Robotics*, Kyoto, Japan, 2011, pp. 245–250.

[7] Y. Iwano, K. Osuka, and H. Amano, "Evaluation of rescue support stretcher system," in *2012 IEEE RO-MAN: The 21st IEEE International*

Symposium on Robot and Human Interactive Communication, Paris, France, 2012, pp. 245–250.

[8] A. Williams, B. Sebastian, and P. Ben-Tzvi, "A robotic head stabilization device for medical transport," *Robotics*, vol. 8, no. 1, p. 23, 2019.

[9] R. P. Saputra and P. Kormushev, "Casualty detection from 3D point cloud data for autonomous ground mobile rescue robots," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics*, Philadelphia, PA, USA, 2018, pp. 1–7.

[10] R. P. Saputra, N. Rakicevic, and P. Kormushev, "Sim-to-real learning for casualty detection from ground projected point cloud data," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Macau, China, 2019, pp. 3918–3925.

[11] R. P. Saputra and P. Kormushev, "ResQbot: A mobile rescue robot for casualty extraction," in *2018 ACM/IEEE International Conference on Human-Robot Interaction*, Chicago, IL, USA, 2018, pp. 239–240.

[12] R. P. Saputra and P. Kormushev, "ResQbot: A mobile rescue robot with immersive teleperception for casualty extraction," in *Towards Autonomous Robotic Systems (TAROS)*, M. Giuliani, T. Assaf, and M. E. Giannaccini, Eds. Cham: Springer, 2018, pp. 209–220.

[13] J. A. Rossiter, *Model-based predictive control: a practical approach*. CRC press, 2017.

[14] J. B. Rawlings and D. Q. Mayne, *Model predictive control: Theory and design*. Nob Hill Pub., 2009.

[15] D. Gu and H. Hu, "A stabilizing receding horizon regulator for nonholonomic mobile robots," *IEEE Trans. Robot.*, vol. 21, no. 5, pp. 1022–1028, 2005.

[16] H. N. Huynh, O. Verlinden, and A. V. Wouwer, "Comparative application of model predictive control strategies to a wheeled mobile robot," *J. Intell. Robot Syst.*, vol. 87, no. 1, pp. 81–95, 2017.

[17] Z. Li, J. Deng, R. Lu, Y. Xu, J. Bai, and C.-Y. Su, "Trajectory-tracking control of mobile robot systems incorporating neural-dynamic optimized model predictive approach," *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 46, no. 6, pp. 740–749, 2015.

[18] S. Yu, Y. Guo, L. Meng, T. Qu, and H. Chen, "MPC for path following problems of wheeled mobile robots," *IFAC-PapersOnLine*, vol. 51, no. 20, pp. 247–252, 2018.

[19] N. Hirose, R. Tajima, and K. Sukigara, "MPC policy learning using DNN for human following control without collision," *Adv. Robot*, vol. 32, no. 3, pp. 148–159, 2018.

[20] M. Neunert, C. de Crousaz, F. Furrer, M. Kamel, F. Farshidian, R. Siegwart, and J. Buchli, "Fast nonlinear model predictive control for unified trajectory optimization and tracking," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, Sweden, 2016, pp. 1398–1404.

[21] T. T. Ribeiro and A. G. Conceição, "Nonlinear model predictive visual path following control to autonomous mobile robots," *J. Intell. Robot Syst.*, vol. 95, no. 2, pp. 731–743, 2019.

[22] H. Fukushima, K. Kon, and F. Matsuno, "Model predictive formation control using branch-and-bound compatible with collision avoidance problems," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1308–1317, 2013.

[23] J. Engsberg, J. Standeven, T. Shurtleff, J. Tricamo, and W. Landau, "Spinal cord and brain injury protection: testing concept for a protective device," *Spinal cord*, vol. 47, no. 8, pp. 634–639, 2009.

[24] "Head injury criteria tolerance levels." accessed: 2020. [Online]. Available: http://www.eurailsafe.net/subsites/operas/HTML/Section3/Page3.3.1.4.htm

[25] R. P. Saputra, E. Rijanto, and H. M. Saputra, "Trajectory scenario control for the remotely operated mobile robot lipi platform based on energy consumption analysis," *Int. J. Appl. Eng. Res.*, vol. 7, no. 8, pp. 851–866, 2012.

[26] Z.-P. JIANGdagger and H. Nijmeijer, "Tracking control of mobile robots: A case study in backstepping," *Automatica*, vol. 33, no. 7, pp. 1393–1399, 1997.

[27] G. Oriolo, A. De Luca, and M. Vendittelli, "Wmr control via dynamic feedback linearization: design, implementation, and experimental validation," *IEEE Trans. Control Syst. Technol.*, vol. 10, no. 6, pp. 835–852, 2002.

[28] D. Chwa, "Sliding-mode tracking control of nonholonomic wheeled mobile robots in polar coordinates," *IEEE Trans. Control Syst. Technol.*, vol. 12, no. 4, pp. 637–644, 2004.

[29] G. Indiveri, "Kinematic time-invariant control of a 2D nonholonomic vehicle," in *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No. 99CH36304)*, vol. 3, Phoenix, AZ, USA, 1999, pp. 2112–2117.

[30] M. W. M. Said, "Optimization based solutions for control and state estimation in non-holonomic mobile robots: stability, distributed control, and relative localization," Ph.D. dissertation, Faculty of Engineering and Applied Science Memorial University of Newfoundland, St. John's, Newfoundland, Canada, December 2018.

[31] T. M. Howard, C. J. Green, and A. Kelly, "Receding horizon model-predictive control for mobile robot navigation of intricate paths," in *Field and Service Robotics*. Berlin, Heidelberg: Springer, 2010, pp. 69–78.

[32] R.-E. Precup, R.-C. David, E. M. Petriu, A.-I. Szedlak-Stinean, and C.-A. Bojan-Dragos, "Grey wolf optimizer-based approach to the tuning of pi-fuzzy controllers with a reduced process parametric sensitivity," *IFAC-PapersOnLine*, vol. 49, no. 5, pp. 55 – 60, 2016, 4th IFAC Conference on Intelligent Control and Automation SciencesICONS 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405896316302853

[33] G. Rigatos, P. Siano, D. Selisteanu, and R. E. Precup, "Nonlinear optimal control of oxygen and carbon dioxide levels in blood," *Intell Ind Syst*, vol. 3, no. 2, pp. 61–75, 2017. [Online]. Available: https://doi.org/10.1007/s40903-016-0060-y

[34] S. Preitl, R.-E. Precup, Z. Preitl, S. Vaivoda, S. Kilyeni, and J. K. Tar, "Iterative feedback and learning control. servo systems applications," *IFAC Proceedings Volumes*, vol. 40, no. 8, pp. 16–27, 2007, 1st IFAC Workshop on Convergence of Information Technologies and Control Methods with Power Plants and Power Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1474667015326318

[35] G. Wen, S. S. Ge, C. L. P. Chen, F. Tu, and S. Wang, "Adaptive tracking control of surface vessel using optimized backstepping technique." *IEEE Trans Cybern*, vol. 49, no. 9, pp. 3420–3431, 2019. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/29994688/

[36] M. Zanon, A. Boccia, V. G. S. Palma, S. Parenti, and I. Xausa, *Direct Optimal Control and Model Predictive Control*. Cham: Springer International Publishing, 2017, pp. 263–382. [Online]. Available: https://doi.org/10.1007/978-3-319-60771-9_3

[37] L. Huang, W. Ning, and Z. Jin-Hui, "Multiobjective optimization for controller design," *Acta Automatica Sinica*, vol. 34, no. 4, pp. 472–477, 2008.

[38] D. De Vito and R. Scattolini, "A receding horizon approach to the multiobjective control problem," in *2007 46th IEEE Conference on Decision and Control*. IEEE, 2007, pp. 6029–6034.

[39] A. Bemporad and D. M. de la Peña, "Multiobjective model predictive control," *Automatica*, vol. 45, no. 12, pp. 2823–2830, 2009.

[40] J. Zhang, M. Norambuena, L. Li, D. Dorrell, and J. Rodriguez, "Sequential model predictive control of three-phase direct matrix converter," *Energies*, vol. 12, no. 2, p. 214, 2019.

[41] C. Rekik, M. Jallouli, and N. Derbel, "Optimal trajectory of a mobile robot using hierarchical fuzzy logic controller," *Int. J. Comput. Appl. Technol.*, vol. 53, no. 4, pp. 348–357, 2016.

[42] M. Easley, M. Hosseinzadehtaher, A. Y. Fard, M. B. Shadmand, and H. Abu-Rub, "Computationally-efficient hierarchical optimal controller for grid-tied cascaded multilevel inverters," in *2019 IEEE Energy Conversion Congress and Exposition (ECCE)*. IEEE, 2019, pp. 219–224.

[43] C. Isik and A. M. Meystel, "Pilot level of a hierarchical controller for an unmanned mobile robot," *IEEE Journal on Robotics and Automation*, vol. 4, no. 3, pp. 241–255, 1988.

[44] K. Moore and N. Flann, "Hierarchical task decomposition approach to path planning and control for an omni-directional autonomous mobile robot," in *Proceedings of the 1999 IEEE International Symposium on Intelligent Control Intelligent Systems and Semiotics (Cat. No. 99CH37014)*. IEEE, 1999, pp. 302–307.

[45] Y. Hasegawa and T. Fukuda, "Learning method for hierarchical behavior controller," in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, vol. 4. IEEE, 1999, pp. 2799–2804.

[46] D.-J. Kim, K.-H. Park, and Z. Bien, "Hierarchical longitudinal controller for rear-end collision avoidance," *IEEE Trans. Ind. Electron.*, vol. 54, no. 2, pp. 805–817, 2007.

[47] A. Melendez, O. Castillo, A. Alanis, and J. Soria, "Reactive and tracking control of a mobile robot in a distributed environment using fuzzy logic," in *International Conference on Fuzzy Systems*. IEEE, 2010, pp. 1–5.

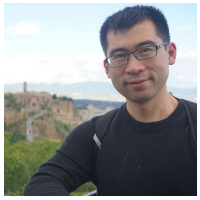
[48] A. Meléndez and O. Castillo, "Hierarchical genetic optimization of the fuzzy integrator for navigation of a mobile robot," in *Soft Computing Applications in Optimization, Control, and Recognition*. Springer, 2013, pp. 77–96.

[49] F. Abdessemed, M. Faisal, M. Emmadeddine, R. Hedjar, K. Al-Mutib, M. Alsulaiman, and H. Mathkour, "A hierarchical fuzzy control design for indoor mobile robot," *Int. J. Adv. Robot. Syst.*, vol. 11, no. 3, p. 33, 2014.
[50] T. T. V. Nguyen, M. D. Phung, and Q. V. Tran, "Behavior-based navigation of mobile robot in unknown environments using fuzzy logic and multi-objective optimization," *Int. J. Control. Autom. Syst.*, vol. 10, no. 1, pp. 349–364, 2017.
[51] R. V. Cowlagi and P. Tsiotras, "Hierarchical motion planning with dynamical feasibility guarantees for mobile robotic vehicles," *IEEE Trans. Robot.*, vol. 28, no. 2, pp. 379–395, 2011.
[52] P. Falcone, F. Borrelli, H. E. Tseng, J. Asgari, and D. Hrovat, "A hierarchical model predictive control framework for autonomous ground vehicles," in *2008 American Control Conference*, Seattle, WA, USA, 2008, pp. 3719–3724.
[53] M. Ibrahim, J. Matschek, B. Morabito, and R. Findeisen, "Hierarchical model predictive control for autonomous vehicle area coverage," *IFAC-PapersOnLine*, vol. 52, no. 12, pp. 79–84, 2019.
[54] C. Branca and R. Fierro, "Hierarchical optimization strategies for deployment of mobile robots," *Int. J. Intell. Control Syst., Special Issue on Swarm Robotics*, vol. 11, no. 3, pp. 141–153, 2006.
[55] A. Agostini, M. Saveriano, D. Lee, and J. Piater, "Manipulation planning using object-centered predicates and hierarchical decomposition of contextual actions," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 5629–5636, 2020.
[56] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3D motion estimation," in *2011 IEEE International Symposium on Safety, Security and Rescue Robotics*, Kyoto, Japan, November 2011, pp. 155–160.
[57] S. Kohlbrecher and J. Meyer, "hector_slam," 2020. [Online]. Available: https://github.com/tu-darmstadt-ros-pkg/hector_slam
[58] B. Lindqvist, S. S. Mansouri, A.-a. Agha-mohammadi, and G. Nikolakopoulos, "Nonlinear mpc for collision avoidance and control of uavs with dynamic obstacles," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6001–6008, 2020.
[59] M. Kamel, J. Alonso-Mora, R. Siegwart, and J. Nieto, "Robust collision avoidance for multiple micro aerial vehicles using nonlinear model predictive control," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 236–243.
[60] E. Small, P. Sopasakis, E. Fresk, P. Patrinos, and G. Nikolakopoulos, "Aerial navigation in obstructed environments with embedded nonlinear model predictive control," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3556–3563.
[61] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Math. Prog. Comp.*, vol. 11, pp. 1–36, 2019.
[62] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, Lake Tahoe, NV, USA, 2012, pp. 2951–2959.
[63] F. Nogueira, "Bayesian Optimization: Open source constrained global optimization tool for Python," 2014–. [Online]. Available: https://github.com/fmfn/BayesianOptimization

RONI PERMANA SAPUTRA received the B.Eng. degree in mechanical engineering from Gadjah Mada University (UGM), Indonesia and the M.Eng.Sc. degree in mechanical engineering (with mechatronics specialisation) from the University of New South Wales (UNSW), Australia. He is currently a Ph.D. student in the Robot Intelligence Lab, Imperial College London, United Kingdom. His research interests include autonomous mobile robot, mobile robot navigation, model predictive control, and robot perception. Currently, he is also part of the research group of mechatronics, Research Centre for Electrical Power and Mechatronics, Indonesian Institute of Sciences (LIPI), Indonesia.

NEMANJA RAKICEVIC is a Ph.D. student in the Robot Intelligence Lab, Imperial College London. He holds a B.Sc. degree in Mechatronics from the Faculty of Technical Sciences, University of Novi Sad and a M.Sc. degree from the double-degree EMARO (European Masters on Advanced Robotics) program. His research interests lie at the intersection of artificial intelligence and robotics, more specifically representation learning, policy search, quality-diversity and deep reinforcement learning, applied to continuous control tasks.

DIGBY CHAPPELL is currently a PhD student at Imperial College London, researching robotics and artificial intelligence. He received his MEng and BA degrees in Engineering from the University of Cambridge, where he specialised in information engineering. His interests lie in researching how robotics can be used to improve healthcare in areas such as prosthetics, medical training, and emergency response.



KE WANG is a PhD student at the Robot Intelligence Lab, Imperial College London. He received his BSc degree in Automotive Engineering at Tongji University, Shanghai, China in 2013 and his Msc degree in Robotics, Cognition and Intelligence from the Technical University of Munich (TUM) in 2016. Ke leads the SLIDER project, which intends to build a knee-less, straight legged bipedal walking robot. In this project he focuses on using optimal control to generate agile movements for bipedal robots. His main research interest is control of legged robots.

DR PETAR KORMUSHEV is a Lecturer (Assistant Professor) in Robotics at the Dyson School of Design Engineering, Imperial College London, UK. He is also the founder and director of the Robot Intelligence Lab (http://www.imperial.ac.uk/robot-intelligence/), and an Academic Fellow of the Data Science Institute at Imperial College London. He holds a PhD in Computational Intelligence from Tokyo Institute of Technology, an MSc in Artificial Intelligence, and an MSc in Bio- and Medical Informatics. Previously, Dr Kormushev was a visiting Senior Research Fellow at King's College London, and a Research Team Leader at the Advanced Robotics department of the Italian Institute of Technology (IIT). Dr Kormushev's research focus is on the application of machine learning algorithms to robotics, especially reinforcement learning for intelligent robot behaviour.