Resource-aware Estimation and Control for Edge Robotics: a Set-based Approach

Dimitrios Spatharakis*, Marios Avgeris*, Nikolaos Athanasopoulos†, Dimitrios Dechouniotis*
Symeon Papavassiliou*
* *School of Electrical and Computer Engineering, National Technical University of Athens, Greece*
†*School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast, Northern Ireland, UK*
{dspatharakis, mavgeris, ddechou}@netmode.ntua.gr, n.athanasopoulos@qub.ac.uk, papavass@mail.ntua.gr

***Abstract*—The evolution of the Industrial Internet of Things (IIoT) and Edge Computing enables resource-constrained mobile robots to offload the computationally intensive localization algorithms. Naturally, utilizing the remote resources of an edge server to offload these tasks, encounters the challenge of a joint co-design in communication, control, estimation and computing infrastructure. We introdce a set-based estimation offloading framework, for the specific case of the navigation of a unicycle robot towards a target position. The robot is subject to modeling and measurement uncertainties, and the estimation set is calculated using overapproximation techniques that alleviate additional computations. A switching set-based control mechanism provides accurate navigation and triggers more precise estimation algorithms when needed. To guarantee the convergence of the system and optimize the utilization of remote resources, a utility-based offloading mechanism is designed, which takes into account both the dynamic network conditions and the available computing resources at the network edge. The performance of the proposed framework is demonstrated through simulations and comparison with alternative offloading schemes.**

***Index Terms*—Mobile Robotics, Cyber-Physical Systems, Computational Offloading, Set Estimation, Edge Robotics.**

I. Introduction

Smart manufacturing leverages the key-enabling 5G technologies to modernize and improve the performance of production workflows. Adopting the Industrial Internet of Things (IIoT) paradigm, a factory floor is equipped with various interconnected devices, i.e., robots, sensors and actuators, which can be dynamically configured [1]. Furthermore, the Edge Computing service delivery model and network softwarization concepts provide the essential computing resources for deploying end-to-end network services, enabling the interplay between industrial equipment and applications of different vendors [2]. These novel technologies facilitate the formulation of Cyber-Physical Systems (CPS) that require automated decisions in the sense-compute-actuate cycle.

Under this setting, the dynamic support of CPS depends heavily on the provisioning of the right piece of data to the right computing entity in a timely manner [3]. This introduces new research and design challenges concerning data processing, offloading decisions, resource allocation and controllers' design [4]. Regarding data processing, Machine Learning and Deep Learning techniques have been widely adopted to train models, either locally or remotely, for predicting the behavior of the IIoT devices [5]. With reference to the offloading decision, it is beneficial only if reliable and low-latency wireless connectivity is available. This generates the need for the network status to be taken into account. With respect to resource allocation, the IIoT-based applications are computationally intensive and require a significant amount of resources for task execution. This imposes the design of dynamic resource allocation strategies for the underlying edge and cloud infrastructure, which in turn facilitates the deployment of industrial applications as network services that can be reconfigured on demand. A resource allocation strategy must guarantee the performance requirements of the applications. To this end, the controller synthesis should provide formal closed-loop guarantees and stabilize the industrial process leading to safe and optimal behaviors. Towards successful closed-loop operations, the co-design of Control, Communication and Computing (3C) is required to provide reliable, low-latency and high-performance control for industrial processes.

This article focuses on the case of Edge Robotics [6], which is widely used in 5G industrial verticals. Following the current trend in service delivery, Edge Robotics leverages the computing capabilities of Edge Computing to achieve low-latency communication [7]. In this work, the mobile robot considered is a unicycle, subject to modeling and measurement uncertainties. In the proposed scenario, the robot must solve a state estimation, also called localization, problem and subsequently a trajectory tracking problem. Under this setting, an offloading mechanism is available for transmitting the sensing data of the localization procedure to an edge server for further processing. Following recent works in the literature, [8], a set-based estimation approach is considered, as our main concern is to provide deterministic guarantees on the robot's estimated pose and subsequently guarantee convergence towards a target waypoint. Although existing elegant solutions for control of unicycle robots exist, e.g., [9], [10], incorporating set-based estimation methods into the controller design to provide real-time navigation, is quite challenging. The proposed algorithm aims to provide a set-based estimation offloading mechanism in the context of Edge Robotics. Under this complex scenario, the fundamental

trade-off between performance and consumed resources is investigated, along with the conditions that guarantee the system's convergence.

A. Related Work

This section presents the most recent related studies in the literature that discuss the deployment of robotic applications, such as path planning, localization and obstacle avoidance. Depending on whether local or remote resources are utilized, these studies are classified into two groups; (i) the onboard sensor-based approaches and (ii) the offloading-based ones.

Due to limited computing resources, the onboard sensor-based approaches aim to provide lightweight solutions for robotic applications. Bajcsy et al. [11] proposed a safe navigation framework for autonomous vehicles moving in a priory unknown static environments under the assumption that the sensors work perfectly within their ranges. This framework was based on Hamilton Jacobi reachability analysis. Due to the computationally expensive nature of this analysis, the authors proposed an algorithm that uses only new measurements to update the safety set and can be executed in an online fashion. On the other hand, the authors in [12] focused on the path planning of autonomous vehicles that are able to maneuver on the road. The path planning problem was formulated as a nonlinear optimization problem and two Model Predictive Control solutions were designed for the lane selection and collision avoidance problems respectively. Miller et al. [13] proposed a controller synthesis algorithm for path planning in dynamic and partially unknown environments. The proposed vehicle system consisted of three subsystems: (i) a perception subsystem that provided a free space prediction around the vehicle, (ii) a plant-controller subsystem that guided the vehicle to specific waypoints and (iii) the planner subsystem that produced safe reference trajectories using Mixed Integer Linear Programming. Finally, the authors in [14] proposed a multi-robot collaborative localization framework, where followers assisted the self-localization of the leaders in a time-varying measurement topology. A binomial regulation function was used to describe the loss or resurgence of an observation. Then, a centralized extended Kalman filter was implemented for estimation purposes.

The offloading-based studies leverage the network and computing capabilities of edge servers to execute remotely navigation or localization algorithms. The authors in [15], similar to this work, presented a two-layer architecture for the realization of a visual-based Simultaneous Localization And Mapping application (SLAM) for tracking. They proposed a lightweight version of the computationally-intensive visual SLAM algorithm that is more suitable for resource-constrained mobile devices. On the other hand, a more precise variation of the visual SLAM is deployed on an edge server in proximity. The mobile devices transfer keyframes of a video for further processing at the edge side, only when necessary, i.e., when the feature points between two consecutive images are not similar. The offloading decision also considers the network conditions.

Chinchali et al. [16] proposed network offloading for Cloud Robotics. The offloading problem was formulated as a Markov Decision Process (MDP), where an autonomous system updated the offloading decision at every time interval. Then, Deep Reinforcement Learning (DRL) was used for solving the offloading problem taking into account the diverse network conditions and the trade-off between local and remote computation. On the other hand, the authors in [17] focused on data representation for task-centric communication rather than addressing the decision making problem of when to offload. Based on DRL, a robot encoder compressed and transmitted concise representations instead of raw data, while a server decoder generated a reconstructed estimation of the raw sensory input. Then, one of the pre-trained task modules was used to predict object locations and classes. Spatharakis et al. [7] proposed a switching offloading mechanism for robot path planning and localization. In that work, both services could be executed either locally, on the mobile robot, or remotely on an edge server. The offloading problem was formulated as a switching model that balanced the trade-off between navigation accuracy and mission duration. The offloading decision took into account both the robot pose uncertainties and the resource availability at the server side. In a different manner, the authors in [18] proposed a symbiotic robotic network for task offloading in the factory floor. Based on their vicinity, the robots formed clusters where members could offload tasks to each other. Additionally, a reward-based feedback task offloading mechanism was proposed to support delay-sensitive applications. Based on these rewards, each node had a social repute score which was used to select the appropriate node to offload the tasks and for the election of the cluster head.

B. Contributions & Outline

In a smart factory environment, the robots are equipped with sensors that can provide estimations of the robot's pose (i.e., two-dimensional location and orientation) [19]. Although employing onboard sensor-based localization yields quick results, they are known to be prone to accumulative errors, especially when it comes to long trajectories [20]. Thus, more sophisticated yet computationally intensive techniques are usually required to increase the localization accuracy. However, as discussed in [6], mobile robotic agents have limited computing capabilities. To overcome these limitations, an Edge Computing infrastructure in the robots' proximity can be leveraged to undertake the computationally intensive localization tasks. The communication between the robots and the servers can be performed through wireless access points located also on the operating floor, while 5G connectivity can be applied as well if available.

Aiming to analyze the above trade-offs and propose novel control co-design strategies that guarantee the correct behavior of such a system, this article presents a control co-design methodology for mobile robot navigation. Specifically, we consider a unicycle operating on a smart factory floor in an Industry 4.0 application, able to move independently without following predefined trajectories. The robot, equipped with cameras and odometry sensors, navigates from a starting to a target position, to complete a given mission (e.g., an automated

storage/retrieval). In this context, the trade-off between the accuracy of the navigation and the mission duration is investigated, according to the mission's characteristics.

The main contributions of our work that differentiate it from the rest of the literature, are summarized as follows:

1) A 3C co-design for CPS is introduced where a unicycle-type mobile robot utilize both onboard and remote resources to execute the computationally intensive tasks of an Industry 4.0 application that requires navigation in a factory floor. Two fundamental problems are jointly tackled: (i) the synthesis of controllers that dictate the motion of the unicycle robot in this path planning problem and (ii) an offloading strategy to compensate the uncertainty of the local estimation techniques with the more accurate remote ones, while finding the balance between navigation accuracy and mission duration. In the context of event-triggered control and following well-acknowledged works, e.g., [21] and [22], we introduce a framework that incorporates the network and computation availability together with stability-preserving conditions. To the best of our knowledge, this is the first work that introduces such a solution.
2) Novel controllers are designed to satisfy the mission's hard constraint, i.e., to ensure the convergence of the mobile robot's navigation to a target set. A unicycle kinematic model is assumed for the robot's dynamics, while its movement is broken down into two parts, i.e., rotational and translational. We note that, although there are works in the literature providing elegant robust controllers in the presence of uncertainties, e.g., [10], [23], our proposed method allows us to deal with three distinct challenges, namely, (i) uncertainties/disturbances acting on the dynamics of the third state, (ii) efficiently applying set-based estimation using odometry measurements, (iii) guaranteeing convergence of the closed-loop system to a target waypoint. To further alleviate the computational strain from the resource-constrained platform, approximate computations are employed to locally estimate the robot's pose. Subsequently, stabilizing state feedback control mechanisms are applied to both movements, to guarantee convergence to the target region.
3) A utility function-based decision making process is properly formulated to undertake the computational offloading strategy. This strategy dictates which of the two different available localization techniques are used; an error-prone, for example, odometry-based localization, executed locally on the robot and the accurate vision-based localization method that requires a significant amount of computing resources and is executed on the edge server. Apart from the navigation's quality, which is acquired by the controllers' outputs, this process also takes into consideration the networking and edge computing resource availability to regulate the trade-off between navigation accuracy and mission duration, based on the performance requirements of the deployed application.
4) A series of experiments are performed to evaluate the performance of the proposed CPS in terms of navigation precision and mission duration. The evaluation indicates that the desirable solution concerning the preference of the two localization algorithms is a mixed strategy employing both of them. Using exclusively a locally produced estimation, is not sufficient to provide high enough accuracy, while constantly seeking for a more precise estimation from the remotely executed algorithm adds significant overhead in mission duration. Furthermore, a detailed comparative evaluation with alternative offloading schemes demonstrates our framework's benefits, as well as its adaptability to the application's specific requirements.

The remainder of the paper is organized as follows: in Section II the problem under consideration is described and defined, while in Section III the system dynamics and the approximation analysis are presented. In Section IV, the details of control design and the guarantee of convergence are introduced, while Section V provides the details on the offloading decision mechanism. In Section VI, the mechanism performance is evaluated and indicative numerical results on the robot's navigation are presented. Finally, Section VII concludes the paper and highlights potential future work.

II. Problem Definition

Formally introducing the overall problem, let $x \in \mathbb{R}^3$ be the pose (states) and $u \in \mathbb{R}^2$ the control actions. Let us denote by $\mathcal{X}_t \subset \mathbb{R}^3$ the estimation set for $x(t)$ for each time instant t. Specifically, $\mathcal{X}_t$ contains all the possible states the robot can reach starting from an initial state $x(0) = x_0 \in \mathbb{R}^3$, with an initial uncertainty $\mathcal{X}(0) = \mathcal{X}_0 \in \mathbb{R}^3$. The problem of estimating a conservative approximation of $\mathcal{X}_t$ is a rather challenging computational problem for general dynamics and constrained environments [24, Chapter 10]. In our setting, we assume that two localization algorithms are available; (i) a fast, locally-executed one, providing an unreliable estimation while moving and (ii) a time-consuming, however more precise one, executed remotely on an edge server. Furthermore, we consider we have the option of invoking either of the two algorithms above; in fact, the challenge addressed in this work is to provide a stabilizing tracking controller and the corresponding offloading decision mechanism. We let $\kappa(t)$ be a switching signal that denotes which estimation algorithm is used at each time instant. Then, the twofold problem lies on the specification of:

a) the robot's respective control actions $u(t)$, under the current estimation set, such that the robot reaches and remains after finite time ϵ-close to the target position $x^\star$, i.e., there exists a t_f such that $||x(t) - x^\star|| \leq \epsilon$, for all $t \geq t_f$ and
b) the offloading control action $O(t) = h(q_o)$, where q_o denotes a utility function that incorporates the current navigation quality, network conditions and the edge computing resources availability.

TABLE I: Summary of the key notation.

Symbol	Interpretation
$x_1(t), x_2(t)$	Robot's position at time t
$x_3(t)$	Robot's orientation at time t
$u_1(t), u_2(t)$	Control actions at time t
$x^\star$	Target position
T	Time period - sampling time
$\sigma(t)$	Model switching signal
$\kappa(t)$	Offloading switching signal
$\delta(t)$	Heading error - disturbance at time t
$\mathcal{D}$	Heading error domain
$y_1^\star(t), y_2^\star(t)$	Measurements at time t
$w_1(t), w_2(t)$	Robot sensors' errors at time t
$\mathcal{W}$	Measurement errors domain
$\mathcal{X}_o$	Initial Estimation Set
$\mathcal{X}_t$	Estimation set at time t
$\mathcal{X}_t^{1:2}$	Estimation set of position at time t
$\mathcal{X}_t^3$	Estimation set of orientation at time t
$\hat{\mathcal{X}}_t$	Overapproximated Estimation set at time t
$Vol(\hat{\mathcal{X}}_t)$	Volume of $\hat{\mathcal{X}}_t$ set at time t
$\mathcal{Z}_t$	One-step Reachable set at time t
$\hat{\mathcal{Z}}_t$	Overapproximated One-step Reachable set at time t
$\mathcal{C}_t$	Compatibility set at time t
$d(\cdot,\cdot)$	Euclidean distance between two points
$V(\cdot,\cdot)$	Distance from a set to a point
$\Phi(\cdot,\cdot)$	Orientation-target orientation incline
M	Set of states where translation motion is allowed
ϵ	Acceptable distance from target position
$\mathcal{G}$	Goal set
$x^{repr}(t)$	Representative point of $\hat{\mathcal{X}}_t$ at time t
$e_{ul}(d)$	Transmission time of offloaded task at time t
$e_{comp}(t)$	Computation time of offloaded task at time t
$q_o(t)$	Utility function value at time t
$O(t)$	Offloading strategy at time t
c_1, c_2	Coefficients of the Utility function

Under this setting, a CPS is formulated by the mobile robot and the edge infrastructure, which collaborate to satisfy the requirements of a navigation application.

In the context of this work, the target position is dictated by an external path planning algorithm, for example as in [7], under a smart industry application. Moreover, this consideration allows us to ignore the case of obstacles located in the factory floor, as the path planning algorithm provides a safe trajectory. Table I summarizes the key notation used throughout the article.

III. Dynamics and Approximation Analysis

In this section, we introduce our proposed method to estimate the robot's pose under specific robot dynamics and uncertainties, a problem which is considered to be challenging throughout the literature [24].

A. System Dynamics

As mentioned before, for the robot's movement, we consider the unicycle kinematic model, also equivalent via an affine transformation to the differential drive dynamics. This model assumes that the robot has two wheels that can rotate at different rates, allowing motion by changing the orientation and the position either separately or simultaneously. More specifically, the unicycle kinematics are given as follows, with the respective state space representation

$$\begin{cases} \dot{x}_1(t) &= u_1 \cos x_3(t), \\ \dot{x}_2(t) &= u_1 \sin x_3(t), \\ \dot{x}_3(t) &= u_2, \end{cases} \tag{1}$$

where x_1, x_2 refer to the position and x_3 to the orientation of the robot. We consider the control actions $(u_1(t), u_2(t))$ to be piecewise constant since the implementation of the control action is digital and the sampling time T is considered also constant. Moreover, as noted before, in this work, a path planning algorithm has already provided the target position the robot has to reach to complete its mission.

One celebrated approach in the bibliography [25, p. 96] is to manipulate the unicycle model by breaking down and discretizing the motion to two parts, i.e., rotate/adjust the orientation of the robot ("rotational motion") and move forward towards the target ("translational motion").

Hence, two subsystems are defined. The signal $\sigma(t) : \mathbb{R} \to \{0, 1\}$, switches between a translation (S_1, $\sigma(t) = 0$) and a rotation (S_2, $\sigma(t) = 1$), respectively. The systems with the respective state space representation are defined as:

$$S_1 : \begin{cases} \dot{x}_1(t) &= u_1(t) \cos(x_3(t)), \\ \dot{x}_2(t) &= u_1(t) \sin(x_3(t)), \\ \dot{x}_3(t) &= \delta(t), \end{cases} \tag{2}$$

$$S_2 : \begin{cases} \dot{x}_1(t) &= 0, \\ \dot{x}_2(t) &= 0, \\ \dot{x}_3(t) &= u_2(t) + \delta(t). \end{cases} \tag{3}$$

We assume that the effect the rotational motion has on the robot's position is negligible, while a bounded, however unknown, heading non-zero error $\delta(t) \in \mathcal{D} = [\delta_{min}, \delta_{max}]$ is applied to both motions. We should note here that, for the simplified case of zero disturbance, we utilize directly the dynamics derived by integrating both parts of Systems (2), (3), as in this case the x_3 state is now a constant. The modeling of the controllers and the proof of convergence remain the same. At each time step, the robot must decide whether to rotate or move forward to a straight line, by choosing the corresponding model to reach to a target position $x^\star$. Further analysis on the respective controllers is presented in Section IV. Regardless, the estimation set $\mathcal{X}_t$ for the three states grows from the initial estimation $\mathcal{X}_0$, as the robot moves. As in [24, Chapter 10], we propagate this set forward, given the control action $u(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \end{bmatrix}$ with $u_i(t) \in \mathcal{U}_i \subset \mathbb{R}$, $i = 1, 2$. Then, for the selected piecewise-constant control action and for an unknown but constant disturbance $\delta \in \mathcal{D}$, we solve the two subsystems analytically over one time period T

$$S_1 : \begin{cases} x_1(t+1) = x_1(t) + \frac{u_1(t)}{\delta(t)}[\sin(x_3(t+1)) - \sin(x_3(t))] \\ x_2(t+1) = x_2(t) + \frac{u_1(t)}{\delta(t)}[\cos(x_3(t)) - \cos(x_3(t+1))] \\ x_3(t+1) = x_3(t) + T\delta(t), \end{cases} \tag{4}$$

$$S_2 : \begin{cases} x_1(t+1) &= x_1(t), \\ x_2(t+1) &= x_2(t), \\ x_3(t+1) &= x_3(t) + T(u_2 + \delta(t)), \end{cases} \tag{5}$$

where T is the constant sampling time. We define the following mapping functions g_1, g_2 which consist of the dynamics of eq. (4),(5) respectively

$$g_1(x,u,\delta) = \begin{bmatrix} x_1 + \frac{u_1}{\delta}[\sin(x_3+T\delta) - \sin(x_3)] \\ x_2 + \frac{u_1}{\delta}[\cos(x_3) - \cos(x_3+T\delta)] \end{bmatrix},$$
$$g_2(x,u,\delta) = x_3 + T(u_2+\delta)$$

The one-step reachable set $\mathcal{Z}_{t+1}$ is defined next; that is, the set of states that the robot can reach from the estimation set $\mathcal{X}_t$. To reduce complexity, the calculation of the one-step reachable set $\mathcal{Z}_{t+1}$, is decoupled by first computing the one-step reachable set for each state and then calculating the Cartesian product of these sets $\mathcal{Z}_{t+1} = Z^{1:2}_{t+1} \times Z^{3}_{t+1}$, where:

$$\mathcal{Z}^3_{t+1} = \Bigg\{ z \in \mathbb{R} : \Big(\exists x_3 \in \mathcal{X}^3_t, \exists \delta \in \mathcal{D} : \\ z_3 = \begin{cases} x_3 + T\delta, & \text{if } \sigma(t) = 0 \\ g_2(x_3,u,\delta), & \text{if } \sigma(t) = 1. \end{cases} \Big) \Bigg\}. \tag{6}$$

and $\mathcal{Z}^{1:2}_{t+1}$ is computed as the Cartesian product of the propagated states $z_1(t), z_2(t)$, as follows

$$\mathcal{Z}^{1:2}_{t+1} = \Bigg\{ z \in \mathbb{R}^2 : \Bigg(\exists \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \in \mathcal{X}^{1:2}_t, \exists x_3 \in \mathcal{X}^3_t, \exists \delta \in \mathcal{D} : \\ z = \begin{cases} g_1(x_1, u(t), \delta) & \text{, if } \sigma(t) = 0 \\ \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} & \text{, if } \sigma(t) = 1 \end{cases} \Bigg) \Bigg\}. \tag{7}$$

Moreover, as the robot moves, the onboard sensors provide a local pose estimation, $y^\star(t+1)$. Typically, these measurements come from odometry calculations and concern distance travelled and changes in orientation [26, Chapter 5]. Hence, in the context of this work, we consider that the local estimation is of the following form:

$$\begin{bmatrix} y^\star_1(t+1) \\ y^\star_2(t+1) \end{bmatrix} = \begin{bmatrix} \|x_{1:2}(t+1) - x_{1:2}(t)\|^2_2 + w_1(t) \\ x_3(t+1) - x_3(t) + w_2(t) \end{bmatrix}, \tag{8}$$

where $w = \begin{bmatrix} w_1(t) & w_2(t) \end{bmatrix}^\top$ are the sensors' errors for the distance and the shift in orientation measurements respectively, that are unknown, however, bounded, i.e., $w \in \mathcal{W} \subset \mathbb{R}^2$. Then, the compatibility set, $\mathcal{C}_{t+1}$, which includes all the states compatible with the current measurements, is introduced as,

$$\mathcal{C}_{t+1} = \Bigg\{ v \in \mathbb{R}^3 : \Big(\exists w \in \mathcal{W}, \exists x \in \mathcal{X}_t \Big) : \\ \begin{bmatrix} y^\star_1(t+1) - \|v_{1:2} - x_{1:2}\|^2_2 \\ y^\star_2(t+1) - v_3 + x_3 \end{bmatrix} \in \mathcal{W} \Bigg\}. \tag{9}$$

Finally, to complete the calculation of the estimation set, we compute the intersection set of $\mathcal{C}_{t+1}$ with $\mathcal{Z}_{t+1}$, $\mathcal{X}_{t+1}$, which comprises of the output compatible states

$$\mathcal{X}_{t+1} = \mathcal{C}_{t+1} \cap \mathcal{Z}_{t+1}. \tag{10}$$

B. Approximation Analysis

The exact calculations for the one-step reachable set and the compatibility set are challenging and usually computationally intensive [24, Chapter 10]. To alleviate the computational strain from the resource-constrained robot, we employ approximate computations, thus managing to provide fast computations of the estimation set. Specifically, we employ parallelotope overapproximations for the $\mathcal{Z}_{t+1}$ set, computed by eq. (6),(7) and for the $\mathcal{C}_{t+1}$ set computed by eq. (9).

1) Approximation of the one-step reachable set $\mathcal{Z}$: For the problem of computing the one-step reachable set for each state (eq. (6),(7)), a Taylor Model (TM) approximation is invoked, as in [27]. TMs are used to represent flowpipes, i.e., a set of states reachable by continuous dynamics from an initial set within a given time interval. As a result, they can be used to provide guaranteed enclosures to the solutions of ordinary differential equations, often involving non-linear functions, such as the model into consideration. The interested reader may refer to [28] for further analysis of a TM flowpipe construction for non-linear hybrid systems. In the scope of this work, given the non-linear continuous system defined in eq. (2),(3) and the current estimation set $\mathcal{X}_t$, acting as the initial set at each time, we compute the TM flowpipes for each state variable, such that a polynomial overapproximation is computed, given the respective control action $u(t)$. In this way, each overapproximated state variable lies on an interval. Next, to acquire a guaranteed overapproximation of eq. (4), the Cartesian product of the three intervals of the states is computed. As a result, $\mathcal{Z}_{t+1}$ is overapproximated by a parallelotope in the 3-D space, hereinafter, denoted as $\hat{\mathcal{Z}}_{t+1}$ and computed by

$$\hat{\mathcal{Z}}_{t+1} = \hat{\mathcal{Z}}^1_{t+1} \times \hat{\mathcal{Z}}^2_{t+1} \times \hat{\mathcal{Z}}^3_{t+1} = [z_1^{\min}(t+1), z_1^{\max}(t+1)] \times \\ [z_2^{\min}(t+1), z_2^{\max}(t+1)] \times [z_3^{\min}(t+1), z_3^{\max}(t+1)]. \tag{11}$$

The reachable sets for the remaining states are computed similarly. We should note here that different approximation methods for the calculation of the one-step reachable set can be chosen, using for example a Bernstein polynomial basis [29], [30]. Moreover, the exact calculation of the one-step reachable set is known [31] or can be approximated [32], however for recursive calculations, the computation time rises. For our work particularly, we selected the TM approximation due to its popularity and the fact that, as the initial set becomes smaller or the order of the polynomial used for approximation becomes larger [28, Chapter 3.3], the overapproximation becomes more accurate. On the other hand, shorter sampling time results in more accurate approximations, increasing, however, the complexity. The proposed technique is computationally light and complexity-preserving. Hence, it is suitable for calculating the reachable set repeatedly and providing real-time navigation to the resource-constrained mobile robot. The calculation of the estimation set (eq. (10)), i.e. the intersection of the reachable set with the compatibility set, is in principle a non-convex problem. As a result, using a fixed-complexity polytope as a template for the reachable set, alleviates the complexity and reduces computation time, as presented in the next subsection.

2) Approximation of the estimation set $\mathcal{X}$: As stated in [24, Chapter 10], even more challenging is the calculation of the output compatible states. These states are defined by eq. (9). Hence, an interval for where each state lies can be found. Again, a parallelotope approximation $\hat{\mathcal{X}}$ is used to provide an overapproximation of the $\mathcal{X}$ set. Specifically, we aim to further reduce the overapproximation acquired with $\hat{\mathcal{Z}}_{t+1}$, by investigating the compatibility with the acquired measurements $y^\star(t+1)$. Thus, we formulate two optimization problems for each state variable, to find the respective maximum and minimum values of this interval. As a result, the calculation of the interval approximation of the output compatible set can be achieved by solving generally nonlinear

optimization problems. Specifically, to find the maximum attainable value for x_1, given a measurement $y^* \in \mathbb{R}^2$, we solve the following:

$$\begin{aligned} \underset{v_1,v_2,w_1,x_1,x_2}{\text{maximize}} \quad & v_1 && \text{(12a)}\\ \text{subject to} \quad & y_1^* - \|v_{1:2} - x_{1:2}\|_2^2 \in \mathcal{W}^1, && \text{(12b)}\\ & (x_1, x_2) \in \hat{\mathcal{X}}_t^{1:2}, && \text{(12c)}\\ & (v_1, v_2) \in \hat{\mathcal{Z}}_{t+1}^{1:2}. && \text{(12d)} \end{aligned}$$

From the solution of the above problem, the maximum output-compatible value $u_1^{\max}$ for the x_1 state can be obtained. Similarly, to find the minimum output-compatible value for each state, an equivalent optimization problem with different cost functions and the same constraint set, is solved. All such problems are quadratically constrained linear programs. It should be noted that, the output compatible values for x_1, x_2 states must be positive, as we consider that the operating ground corresponds to the first quadrant of a Cartesian plane. Since $\mathcal{W}_1 = [w_1^{\min}, w_1^{\max}]$, with $w_1^{\min} \in \mathbb{R}$, $w_1^{\max} > 0$ being an interval, relation (12b) is equivalent to:

$$\|v_{1:2} - x_{1:2}\|_2^2 \leq y_1^* - w_1^{\min}, \quad \text{(13)}$$

$$\|v_{1:2} - x_{1:2}\|_2^2 \geq y_1^* - w_1^{\max}. \quad \text{(14)}$$

Ineq. (13) contributes with a convex constraint and (14) with a concave constraint respectively, thus, the total constraint set of the optimization problem is non-convex. As a result, the problem cannot be easily solved with efficient convex interior-point methods. Nevertheless, an optimal solution can be found by exploiting the other interval constraints.

First, the solutions for the maximization problem without constraint (14) and the minimization problem without constraint (13), are presented. Subsequently, using Proposition 1, we showcase that the constraints for each problem are redundant. For example, regarding Problem (12a), let us focus on constraint (13): The general solution of the inequality is: $-\sqrt{y_1 - w_1^{\min} - (v_2 - x_2)^2} + x_1 \leq v_1 \leq \sqrt{y_1 - w_1^{\min} - (v_2 - x_2)^2} + x_1$, therefore the maximum positive output-compatible value $v_1^{\max}$ is attained in

$$v_1^{\max} = \min\Big\{v_1^{\max}, x_1^{\max} + \sqrt{y_1^\star - w_1^{\min} - \min_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2}\Big\}, \quad \text{(15)}$$

where $\min_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2$ can be easily calculated, as both v_2, x_2 have known values in intervals. Analogously, for the minimization of v_1, (14) has the following general solution: $v_1 \in (-\infty, -\sqrt{y_1 - w_1^{\max} - (v_2 - x_2)^2} + x_1] \cup [\sqrt{y_1 - w_1^{\max} - (v_2 - x_2)^2} + x_1, \infty)$ and therefore the minimum positive output-compatible value is given by:

$$u_1^{\min} = \max\Big\{v_1^{\min}, x_1^{\min} + \sqrt{y_1^\star - w_1^{\max} - \max_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2}\Big\}, \quad \text{(16)}$$

where, similarly, $\max_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2$ can be easily calculated. We should note here that, in the case where the quantities under the square root are negative for both problems, then they become infeasible and, thus, the maximum and minimum value of v_1 becomes equal to the corresponding value obtained by the one-step reachable set.

Proposition 1. *Let $v_1^{\max}$ be a feasible solution of the maximization problem* (12a) *without constraint* (14) *contributing to the problem and, similarly, let $v_1^{\min}$ be a feasible solution for the respective minimization problem without* (13) *respectively. If $v_1^{\min} \leq v_1^{\max}$, then the two constraints are redundant for solving the respective optimization problems.*

Proof. Let $v_1^{\max}, v_1^{\min} \in \hat{\mathcal{Z}}_{t+1}^{1:2}$ be the two feasible solutions of the maximization problem (12a) without constraint (14) and the corresponding minimization problem without constraint (13), respectively. Then, we want to show that $v_1^{\min} \leq v_1^{\max}$. By construction of the TM overapproximation, we know that $x_1^{\min} \leq x_1^{\max}$. As a result, proving $v_1^{\min} \leq v_1^{\max}$ is equivalent to showing that $x_1^{\min} + \sqrt{y_1^\star - w_1^{\max} - \max_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2} \leq x_1^{\max} + \sqrt{y_1^\star - w_1^{\min} - \min_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2}$. This is also equivalent of proving that $y_1^\star - w_1^{\max} - \max_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2 \leq y_1^\star - w_1^{\min} - \min_{v_2 \in \hat{\mathcal{Z}}_{t+1}^2, x_2 \in \hat{\mathcal{X}}_t^2} (v_2 - x_2)^2$ which is true as both $v_2, x_2, w_1^{\max}, y_1$ are positive. As a result, the two solutions do not intersect and the respective absent constraints do not contribute to the feasible solutions. □

Hence, the solution of a non-convex optimization problem reduces to a simple calculation for each state variable. The minimum and maximum output-compatible values of the x_2 state are attained similarly.

On the other hand, for the output-compatible values of the x_3 state, it suffices to solve two similar linear programs to find the minimum and maximum admissible values. In order to find a maximum value $u_3^{\max}$, the following optimization problem must be solved, derived by eq. (9)

$$\begin{aligned} \underset{v_3,x_3,w_2}{\text{maximize}} \quad & v_3 && \text{(17a)}\\ \text{subject to} \quad & v_3 - x_3 \leq y_2^\star - w_2^{\min}, && \text{(17b)}\\ & v_3 - x_3 \geq y_2^\star - w_2^{\max}, && \text{(17c)}\\ & w_2 \in \mathcal{W}^2, && \text{(17d)}\\ & x_3 \in \hat{\mathcal{X}}_t^3, && \text{(17e)}\\ & v_3 \in \hat{\mathcal{Z}}_{t+1}^3, && \text{(17f)} \end{aligned}$$

where $\mathcal{W}_2 = [w_2^{\min}, w_2^{\max}]$, with $w_2^{\min} \in \mathbb{R}$, $w_2^{\max} > 0$. The minimum output-compatible value $u_3^{\max}$ is attained by the respective minimization problem subject to the same constraints. Finally, having calculated the output-compatible intervals for the three states, e.g., $\hat{\mathcal{X}}_{t+1}^1 \in [u_1^{\min}, u_1^{\max}]$, the overapproximated estimation set $\hat{\mathcal{X}}_{t+1}$ is computed as the Cartesian product of the three sets:

$$\begin{aligned} \hat{\mathcal{X}}_{t+1} &= \hat{\mathcal{X}}_{t+1}^1 \times \hat{\mathcal{X}}_{t+1}^2 \times \hat{\mathcal{X}}_{t+1}^3 \\ &= [u_1^{\min}, u_1^{\max}] \times [u_2^{\min}, u_2^{\max}] \times [u_3^{\min}, u_3^{\max}]. \end{aligned} \quad \text{(18)}$$

Thus, in the context of this work, the estimation set is a parallelotope in the 3-D space. As it will be thoroughly examined in the following section, the low

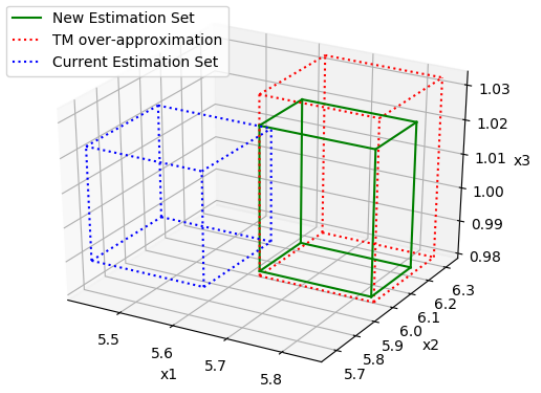


Fig. 1: An example of the overapproximation for the calculation of the next estimation set, $\hat{\mathcal{X}}_{t+1}$.

complexity of the approximation of the estimation set leads to simple calculations for deriving controllers for each motion. Moreover, the volume of the estimation set in the 3-D space is introduced:

$$\text{Vol}(\hat{\mathcal{X}}_{t+1}) = (u_1^{\max} - u_1^{\min})(u_2^{\max} - u_2^{\min})(u_3^{\max} - u_3^{\min}). \quad (19)$$

In Fig. 1, an example of our approximation is demonstrated, for the experiment setting introduced in Section VI. The blue-dashed line denotes $\hat{\mathcal{X}}_t$, while the red-dashed line the overapproximation using the TM technique. Finally, the next estimation set, $\hat{\mathcal{X}}_{t+1}$, is drawn with a green line. Additionally, for the specific example of translational motion, Table II presents the actual overapproximation of our method in numbers. For the calculation of these sets, the following values are used $w_1^{\min} = 0.38$, $w_1^{\max} = -0.22$, $w_2^{\min} = -0.008$, $w_2^{\max} = 0.002$ and $\delta = 0.1$. More details regarding the measurement errors and the disturbance follow in Section VI.

TABLE II: A numerical example of the proposed technique.

i	$\hat{\mathcal{X}}_t^i$	$\hat{\mathcal{Z}}_{t+1}^i$	$\hat{\mathcal{X}}_{t+1}^i$	$u_1(t)$	$y_1^\star$	$y_2^\star$
1	[5.42, 5.64]	[5.62, 5.84]	[5.62, 5.83]	0.38	0.15	0.1
2	[5.67, 6.00]	[5.98, 6.32]	[5.98, 6.20]			
3	[0.98, 1.02]	[0.97, 1.03]	[0.97, 1.02]			

We can notice that the conservative approximation of the estimation set, while taking into account the measurements, affects significantly the result, especially in this nonlinear setting. To sum up, the challenging calculation of the estimation set is handled with overapproximation techniques due to the TM properties and our approach of computing the output-compatible states.

IV. Control Design and Theoretical Guarantee of Convergence

In the previous section, the solution for calculating a conservative approximation of the estimation set was agnostic to the control actions. In this section, we propose a stabilizing state feedback control mechanism, that guarantees convergence to the target region. Let $d(x(t), x^\star) = \|x_{1:2}^\star - x_{1:2}(t)\|_2$ be the distance between two points, specifically between a specific point and the target. The distance between a point and a set is defined as:

$$V(\hat{\mathcal{X}}_t, x^\star) = \max_{x \in \hat{\mathcal{X}}_t} d(x, x^\star). \quad (20)$$

Moreover, let

$$\Phi(\hat{\mathcal{X}}_t, x^\star) = \Big\{\varphi \in \mathbb{R} : (\exists x \in \hat{\mathcal{X}}_t : \varphi = x_3 - \tan^{-1}\left(\frac{x_2 - x_2^\star}{x_1 - x_1^\star}\right))\Big\}, \quad (21)$$

be the set of angles between the robot's current estimation of orientation and the line connecting current robot's position and the target position for $\hat{\mathcal{X}}_t$, simply put, the set of angles of incline towards the target. Finally, let $\mathcal{M}$ be the set of states where translation is performed,

$$\mathcal{M} = \Big\{x \in \mathbb{R}^3 : \Big(\exists u_1 \in \mathbb{R} : l\,|u_1| < 2V(\hat{\mathcal{X}}_t, x^\star)\cos(\delta_M)\Big)\Big\}, \quad (22)$$

where $\delta_m = \max\{|\delta_{min}|, \delta_{max}\}$,

$$l = \sqrt{2({\delta_m}^{-2}(1 - \cos(T\delta_m))},$$
$$\delta_M = \max\{\left|T\delta_{min} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|, \left|T\delta_{max} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|\}$$

The reasoning behind the choice of the $\mathcal{M}$ set is presented in Subsection IV-D. In order to select between performing either translational or rotational motion and reduce the overall computational complexity, only the center of the convex hull of the estimation set is investigated in terms of whether it belongs in the M set. Hereinafter, the center point of the current estimation set, $\hat{\mathcal{X}}_t$, is referred to as the *representative point*, $x^{repr}(t)$. As the estimation set is the Cartesian product of three intervals of the states, the coordinates of the representative point are computed straightforwardly:

$$x^{repr}(t) = (x_1^{repr}(t), x_2^{repr}(t), x_3^{repr}(t)), \quad (23)$$

where $x_i^{repr}(t) = \frac{1}{2}(x_i^{\max}(t) + x_i^{\min}(t))$, $i \in [1, 2, 3]$, is the center of the interval of each state $x_i \in \hat{\mathcal{X}}_t^i$. The translational motion is selected if $x^{repr} \in \mathcal{M}$ for any u_1, otherwise the rotational motion is selected. Moreover, we consider that the robot has reached the target position when $\hat{\mathcal{X}}_t \subseteq \mathcal{G}$, with

$$\mathcal{G} = \{x \in \mathbb{R}^3 : V(\hat{\mathcal{X}}_t, x^\star) \leq \epsilon\}, \quad (24)$$

where ϵ is an acceptable distance from the target position, for the mission to be assumed successful. Note that the robot's orientation when the goal set is reached is not considered important. Since in the proposed modeling the motion is considered either strictly translational or rotational, one control input in each case is nonzero. The overall control strategy that includes the state-dependent switching mechanism can be described by a directed graph of Fig. 2, which illustrates the control automaton. It is noted that the closed-loop system can be described by a linear hybrid automaton with non-convex guard conditions. Nevertheless, we do not use this formalism in order to simplify exposition, especially since the convergence proofs that follow are simple enough not to necessitate the adoption of this powerful modeling approach. Independently of the selected estimation technique, the robot moves as described in the control automaton; specifically (i) decides the motion (signal $\sigma(t)$) according to (22) for

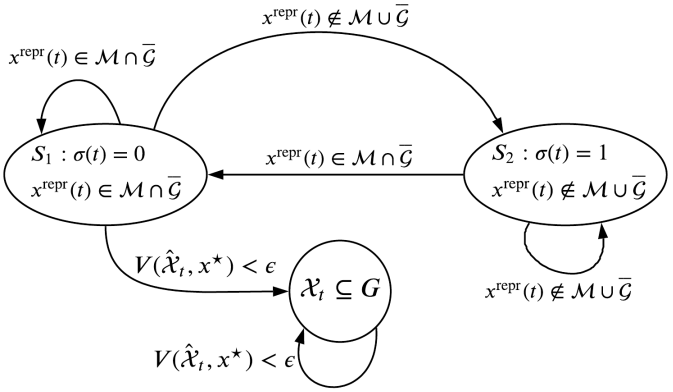


Fig. 2: The control automaton for the robot's motion.

$x^{repr}(t)$, (ii) computes and moves according to the selected control action, (iii) acquires an estimation of its pose using an estimation technique and finally (iv) follows the procedure introduced in the previous section to compute the new estimation set and repeats the process. When the goal set is reached the robot stops. A complete algorithm of the proposed technique is presented in detail in Subsection V-D.

A. Set Controller for Translational Motion

In this subsection, we compute a translational control action that decreases the distance to the target set, i.e.: $V(\hat{\mathcal{X}}_{t+1}, x^\star) < V(\hat{\mathcal{X}}_t, x^\star)$. The control action $u_1(t)$ is computed by the solution of the following optimization problem,

$$\begin{aligned} \underset{\lambda,\lambda_2}{\text{minimize}}\ & \lambda && \text{(25a)}\\ \text{subject to}\ & \lambda^2 \geq 1 + (\lambda_2^2 - 2\lambda_2)\cos^2(\delta_M) && \text{(25b)}\\ & \lambda \in (0,1) && \text{(25c)}\\ & \lambda_2 \in (0,2), && \text{(25d)} \end{aligned}$$

where $\delta_M = \max\{\left|T\delta_{min} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|, \left|T\delta_{max} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|\}$. Then, for the selected λ_2 that minimize λ, we select the following control action

$$l\,|u_1| = \lambda_2 d(x_{1:2}^{repr}, x^\star)\cos(T\delta_M), \tag{26}$$

where $l = \sqrt{2({\delta_m}^{-2}(1-\cos(\delta_m))}$. In this work we consider only forward motion for the unicycle robot; as a result, the positive value of u_1 is selected. Next, for the selected control action $u_1(t)$, the approximation of the one-step reachable set $\hat{\mathcal{Z}}_t$ is computed using eq. (11). Also, as stated in Subsection III-B1, the approximated one-step reachable set $\hat{\mathcal{Z}}_t$, has guaranteed enclosures for the solution of eq. (4) by construction. Let us consider $\hat{\mathcal{Z}}_t^{1:2}$, omitting the orientation plane as the distance function denoted in eq. (20), is defined in the 2-D space. It should be noted here that the following condition for the selected control action is investigating using the reachable set approximation and without taking into account the compatibility set, as this presupposes a control action that is already applied to the robot. To this purpose, the value of eq. (20) can be deduced from the vertices of $\hat{\mathcal{Z}}_t^{1:2}$, since it is a parallelotope overapproximation. Thus, it suffices to investigate whether the distance from the vertices of $\hat{\mathcal{Z}}_t^{1:2}$ towards the target decreases for a given control action. To this purpose, let us denote the vertices of $\hat{\mathcal{Z}}_{t+1}$ as:

$$p_{ij}(t+1) = [z_1^i, z_2^j], \tag{27}$$

where $i = 1, 2$, $j = 1, 2$ indicate the four vertices of $\hat{\mathcal{Z}}_t^{1:2}$. For example, $i = 1$ denotes $z_1^{\min}(t+1)$ and similarly $j = 2$ denotes $z_2^{\max}(t+1)$, where $z_1^{\min}(t+1)$, $z_1^{\max}(t+1) \in \hat{\mathcal{Z}}_{t+1}^1$ and $z_2^{\min}(t+1)$, $z_2^{\max}(t+1) \in \hat{\mathcal{Z}}_{t+1}^2$ as in eq. (11). Consequently,

$$V(\hat{\mathcal{X}}_{t+1}, x^\star) = \max_{i,j\in[1,2]} d(p_{ij}(t+1), x^\star). \tag{28}$$

As a result, given the selected control action $u_1(t)$, the following condition must be investigated to determine if the translational motion decreases the distance of the $\hat{\mathcal{Z}}_t^{1:2}$ set to the target position in one step

$$\max_{i,j\in[1,2]} d(p_{ij}(t+1), x^\star) \leq \lambda' V(\hat{\mathcal{X}}_t, x^\star), \tag{29}$$

where $\lambda' \in (0,1)$. If this condition holds, then the robot implements the translational motion for $u_1(t)$, acquires the measurements and proceeds with the calculation of $\hat{\mathcal{X}}_{t+1}$, as described in Subsection III-B2. Otherwise, eq. (20) is not decreasing and a more precise estimation of the current location is required to approach the target position. Hence, each time the translational motion is allowed, it suffices to find the maximum $\lambda'_{max}(t) \in (0,1)$ for all vertices to guarantee that the distance towards the target is decreasing.

B. Set Controller for Rotational Motion

When the representative point $x^{repr}(t+1)$ does not belong in $\mathcal{M}$, the robot performs a rotational motion. During this, the rotational controller acts to ensure that after one actuation step the translational motion will be allowed. This is essential as naturally the considered function of eq. (20) is not decreasing when the rotational motion is performed.

In this section, an interval of angles of incline $a \in [a_{\min}, a_{\max}]$ of the lines connecting each point in the estimation set to the target position, is specified. The purpose of this is to find, if exists, a rotational control action that shifts the robot's orientation, aiming to satisfy ineq. (22) for $x^{repr}(t+1)$. Since the overapproximated estimation set $\hat{\mathcal{X}}_t$ is a parallelotope in the 3-D space, i.e., the states x_i, $i \in [1,2,3]$ lie in known intervals, then $a_{\min}$ and $a_{\max}$ can be specified by calculating the angles of incline for the lines that connect the vertices of the estimation set, in the 2-D space and the target position, as illustrated in Fig. 3. Consequently, the control action $u_2(t)$ is calculated by solving the following linear problem:

$$\begin{aligned} \min_{u_2}\ & \max_{\hat{\mathcal{X}}_t, a, \delta} (x_3 + \delta - u_2 - a) && \text{(30a)}\\ \text{subject to}\ & x_3 \in \hat{\mathcal{X}}_t^3, && \text{(30b)}\\ & a \in [a_{\min}, a_{\max}], && \text{(30c)}\\ & \delta \in \mathcal{D}, && \text{(30d)}\\ & u_2 \in \mathcal{U}_2, && \text{(30e)} \end{aligned}$$

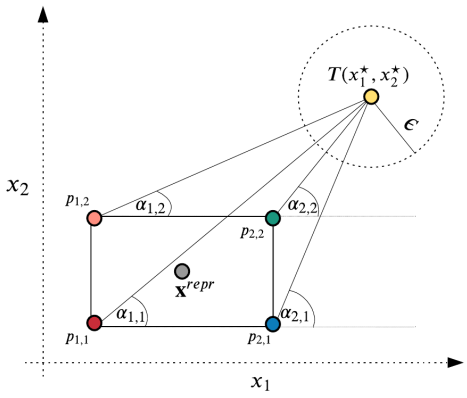


Fig. 3: An illustrated example of the maximum minimum distance and slope between $\hat{\mathcal{X}}_t$ and the target position.

where $a_{\min} = \min\{\tan^{-1}\left(\frac{x_{2,j}(t)-x_2^\star}{x_{1,i}(t)-x_1^\star}\right)\}$ and $a_{\max} = \max\{\tan^{-1}\left(\frac{x_{2,j}(t)-x_2^\star}{x_{1,i}(t)-x_1^\star}\right)\}$, where $i = 1, 2$ and $j = 1, 2$ indicate the four vertices of the estimation set in the 2-D space. For example, $i = 1$ indicates the minimum value of the interval of $x_1(t)$ and similarly $j = 2$ indicates the maximum value for $x_2(t)$. Next, for the selected control action $u_2(t)$, $\hat{\mathcal{Z}}_{t+1}$ is computed by, eq. (11) which also has guaranteed enclosures for the solution of eq. (6) by construction. Similarly to the previous subsection, the compatibility set in not taken into consideration. We define

$$z^{repr}(t+1) = (z_1^{repr}(t+1), z_2^{repr}(t+1), z_3^{repr}(t+1)), \quad (31)$$

where $z_i^{repr}(t+1) = \frac{1}{2}(z_i^{\max}(t+1) + z_i^{\min}(t+1))$, for $i \in [1, 2, 3]$, is the center of the interval of each state $z_i \in \hat{\mathcal{Z}}_{t+1}^i$. It should be noted that when a rotation is performed the robot's position (x_1, x_2) is not affected. As a result, the following condition is examined; whether the shift in orientation endeavors to satisfy ineq. (22) for the representative point $z^{repr}(t+1)$:

$$|u_1| < \frac{2}{l} d(z_{1:2}^{repr}(t+1), x^\star)\cos(\delta_M), \quad (32)$$

where $l = \sqrt{2({\delta_m}^{-2}(1-\cos(T\delta_m)))}$ and $\delta_M = \max\{|T\delta_{min} + \Phi(z^{repr}(t+1), x^\star)|, |T\delta_{max} + \Phi(z^{repr}(t+1), x^\star)|\}$. If $z^{repr}(t+1)$ belongs in $\mathcal{M}$ then the robot performs the rotational motion for $u_2(t)$, acquires the measurements and proceeds with the calculation of $\hat{\mathcal{Z}}_{t+1}$, as described in Subsection III-B2.

To sum up, after the rotational motion, if $x^{repr}(t)$ belongs in $\mathcal{M}$ then the translational motion is allowed in the next step. Hence, the robot performs the translational motion for $u_1(t)$, acquires a measurement and proceeds with the calculation of $\hat{\mathcal{X}}_{t+1}$, as described in Subsection III-B2. Otherwise, a more precise estimation is necessary to proceed towards the target position. One must note that, after rotating with $u_2(t)$, if $z^{repr}(t+1)$ is in $\mathcal{M}$, then, due to the parallelotope overapproximation, it is certain that $x^{repr}(t+1)$ will be in $\mathcal{M}$, as described in Subsection III-B2. That is why translational motion is allowed in the next step.

C. Remote Estimation Technique

It should be clear now that if conditions (29) and (32) are not satisfied for the translational and the rotational motion respectively, then the system can no longer converge to the target position and the switching signal $\kappa(t)$ triggers the precise estimation technique to assist in the navigation

$$\kappa(t) = \begin{cases} 1, & \text{if } \sigma(t) = 0 \text{ and (29) is not satisfied} \\ & \text{or} \\ & \text{if } \sigma(t) = 1 \text{ and (32) is not satisfied,} \\ 0, & \text{otherwise.} \end{cases} \quad (33)$$

Whenever $\kappa(t) = 1$, the localization algorithm described in [19] is invoked remotely. Briefly, this estimation technique relies on a bilateration method using principles of projective geometry. The robot's equipped camera captures images from the area and offloads them into the proximate edge server. There, the localization algorithm analyzes the images to detect landmarks and provide a highly precise estimation regarding the pose of the robot. Thus, this real-time image processing is resource-intensive and time-consuming, even when executed on an edge server. Hence, both the transmission overhead of the images via the access point and the remote processing overhead must be considered to find the right balance between navigation accuracy and mission duration.

In the context of this work, the estimation computed by this technique is considered accurate, without measurement errors. Consequently, after invoking the remote estimation technique, the exact pose of the robot is considered known. This allows us to compute a fine-grained control approach and prove the convergence of the proposed technique. We should note here that, the choice of the selected remote estimation technique is made to showcase a general setting in edge robotics in which computationally intensive algorithms are not to be executed on the robot, but rather to be offloaded on an edge server. Many works exist in the bibliography such as [33], [34] that provide very precise estimations regarding the robot's pose. Moreover, such landmark-based techniques are broadly utilized for indoor localization in the context of Industry 4.0. [35]. It should be emphasized, that the scope of this article is to introduce an offloading strategy between different estimation techniques, seeking a trade-off between mission duration and accuracy and not seeking the best between different methods.

D. Convergence when Constantly Invoking Remote Estimation

In this section, we show that the proposed technique converges to the target position after finite time, when the remote estimation technique is exclusively invoked. Let us assume that at time $t = t^\star$ the remote estimation technique provides a precise estimation to the robot, thus the estimation set is minimized to a point $\mathcal{X}_{t^\star} = \{x^A\}$, as illustrated in Fig. 4. Then, it is straightforward to compute the $\mathcal{Z}_{t+1}$ set using eq. (4) and (5).

The robot initially aligns the orientation accordingly and then proceeds with the translational motion. Another benefit of the precise estimation is that the set-based controller, proposed in Problem (30), considers only one point in order to provide the control action $u_2^\star$. In this way, the robot

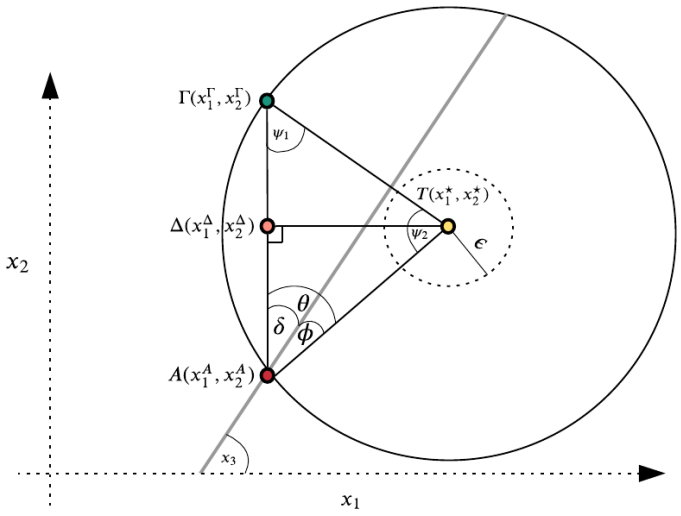


Fig. 4: Convergence example after the rotational motion.

manages to rotate, aligning to (AT), as illustrated in Fig. 4. Thus, the angle of incline towards the target is minimized, specifically $\Phi(\mathcal{Z}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$. As a result, the orientation of the robot $\mathcal{Z}^3_{t^\star+1}$ lies in the interval $\left[\tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{min}, \tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{max}\right]$.

Lemma 1. *Consider the Subsystem* S_2 (5) *and that the remote estimation technique provides an accurate estimation* x^A *at time* $t = t^\star$. *Let* u_2^* *be the control input obtained by the solution of Problem* (30) *applied to* (5) *at time* t^*. *If* $|(T+1)\delta_m| < \frac{\pi}{2}$, *then the translational motion is allowed in the next step, i.e., there exists a control action* $u_1^\star$ *that satisfies ineq.* (22) *for* $\mathcal{Z}_{t^\star+1}$.

Proof. After performing a rotational motion, the robot is still positioned at $x^A_{1:2}$, i.e., $\mathcal{Z}^{1:2}_{t^\star+1} = x^A_{1:2}$, and $\Phi(\mathcal{Z}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$. Moreover, for the translational motion to be allowed, it suffices that $\mathcal{Z}_{t^\star+1} \subseteq \mathcal{M}$. Subsequently, given that $V(\mathcal{Z}_{t^\star+1}, x^\star) = d(x^A, x^\star)$, from ineq. (22) we get:

$$l \ |u_1| < 2d(x^A, x^\star)\cos(\delta_M), \tag{34}$$

where

$$l = \sqrt{2({\delta_m}^{-2}(1 - \cos(T\delta_m))}, \quad \delta_m = \max\{|\delta_{min}|, \delta_{max}\}$$
$$\delta_M = \max\{\left|T\delta_{min} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|, \left|T\delta_{max} + \Phi(\hat{\mathcal{X}}_t, x^\star)\right|\}$$

Under our assumption that $|(T+1)\delta_m| < \frac{\pi}{2}$ and since $\Phi(\mathcal{Z}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$, then l is a positive constant and $0 < \cos(\delta_M) < 1$. As a result, the translational motion is allowed at time $t = t^\star+1$ for all $u_1^\star$ that satisfy ineq. (34) and therefore (22). □

We note that typically the heading error lies in the fraction of a few degrees [36], thus, the condition of Lemma 1 is not particularly conservative since the sampling time T is small.

Proposition 2. *Suppose that the robot rotates at time* $t^\star$. *Then, there exists a control action* $u_1^\star$ *such that* $V(\mathcal{X}_{t+1}, x^\star) \leq \lambda V(\mathcal{X}_t, x^\star)$ *for* $t = t^\star + 1$, $\lambda \in (0, 1)$.

Proof. After rotational motion, by eq. (10) we know that $\mathcal{X}_{t^\star+1} \subseteq \mathcal{Z}_{t^\star+1}$, since the compatibility set is not included in this proof. By Lemma 1, at time $t = t^\star + 1$, the robot is positioned at $x^A_{1:2}$, hence the orientation $\mathcal{X}^3_t$ lies in the interval $\left[\tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{min}, \tan^{-1}\left(\frac{x_2^A - x_2^\star}{x_1^A - x_1^\star}\right) + \delta_{max}\right]$. Moreover, it also holds that $\Phi(\mathcal{X}_t, x^\star) \subseteq \mathcal{D}$ and the translational motion is allowed for all $u_1^\star$ that satisfy ineq. (34). Also, from Subsystem (4), it occurs that the shift in the robot's orientation, produced by the heading error during the translational motion, is equal to $T\delta$. Then, Θ is the set of angles that combine $\Phi(\mathcal{X}_{t^\star+1}, x^\star)$ and this shifts in the orientation, as follows:

$$\Theta(\delta, \Phi(\mathcal{X}_t, x^\star)) = \Big\{\vartheta \in \mathbb{R} : \Big(\exists \delta \in \mathcal{D}, \exists \varphi \in \Phi(\mathcal{X}_{t^\star+1}, x^\star) : \vartheta = T\delta + \varphi\Big)\Big\}. \tag{35}$$

It should be noted that $\theta \in \Theta \subseteq [d_{min} + Td_{min}, d_{max} + Td_{max}]$, since rotational motion is applied first, i.e., $\Phi(\mathcal{X}_{t^\star+1}, x^\star) \subseteq \mathcal{D}$ and d is bounded in $\mathcal{D}$.

Let us also define the circle $C(h, r)$, centered at the target position $x^\star$, $h = T(x_1^\star, x_2^\star)$ and its radius is the distance from the target position $r = (AT) = d(x^A_{1:2}, x^\star)$. If the robot after translational motion is located inside this circle then the distance in one step, is decreasing.

Let $x^\Gamma_{1:2}$ be a point in the circumference of circle C, as illustrated in Fig. 4. By definition, $(A\Gamma)$ chord's length, $d(x^A, x^\Gamma)$, is equal to $2d(x^A, x^\star)\sin(\frac{\psi_2}{2})$. We assume that the robot traverses the $(A\Gamma)$ chord by performing a translational motion with a combined error $\theta \in \Theta$. Using the circle's identities, we get that $\psi_1 = \theta$, as $(A\Gamma T)$ is an isosceles triangle and, thus, $\psi_2 = \pi - 2\theta$. This means that for the $(A\Gamma)$ length we get that $d(x^A, x^\Gamma) = 2d(x^A, x^\star)\cos(\theta)$.

Next, suppose that $x^\Delta_{1:2}$ is a point on $(A\Gamma)$ that the robot can reach in one step, i.e., $x^\Delta_{1:2} \in Z^{1:2}_{t+1}$ for $t = t^\star + 1$. Using basic trigonometric identities, we get that the distance between Δ and the target point $x^\star$, i.e., $d(x^\Delta, x^\star)$, is minimized if Δ is a point on the perpendicular bisector of $(A\Gamma)$, i.e., when its distance from point A is equal to $\frac{d(x^A, x^\Gamma)}{2} = d(x^A, x^\star)\cos(\theta)$. Now, let us assume the general case where

$$d(x^A, x^\Delta) \leq \lambda_2 d(x^A, x^\star)\cos(\theta), \tag{36}$$

where $\lambda_2 \in (0, 2)$. This parametrization guarantees that point Δ is inside the defined circle. The coordinates of x^Δ, can be computed by eq. (4): $x^\Delta = g_1(x^A, u_1, \delta)$. As a result, $d(x^A, x^\Delta) = \|x^\Delta_{1:2} - x^A_{1:2}\|_2$. Using eq. (4) and involving trigonometric identities ineq. (36) becomes:

$$\sqrt{(\frac{u_1}{\delta})^2[2 - 2\cos(T\delta)]} < \lambda_2 d(x^A, x^\star)\cos(\theta). \tag{37}$$

Satisfaction of ineq. (37), which is by assumption true, implies existence of a u_1 satisfying ineq. (34), for $\mathcal{X}_t = x^A$ and for $\theta = \delta_M$. Next, we show that ineq. (37) implies

$$V(\mathcal{X}_{t+1}, x^\star) \leq \lambda V(\mathcal{X}_t, x^\star) \text{ for } t = t^\star + 1. \tag{38}$$

To this purpose, let us assume that, $V(\mathcal{X}_{t+1}, x^\star) = d(x^\Delta, x^\star)$, as point Δ lies arbitrarily in $\mathcal{X}_t$ for $t = t^\star + 1$ Exploiting the law of cosines for the $(A\Delta T)$ triangle we know that for $t = t^\star + 1$

$$V^2(\mathcal{X}_{t+1}, x^\star) = V^2(\mathcal{X}_t, x^\star) + d^2(x^A, x^\Delta) - 2V(\mathcal{X}_t, x^\star)d(x^A, x^\Delta)\cos(\theta) \tag{39}$$

By replacing eq. (39) to eq. (38), then

$$V^2(\mathcal{X}_t, x^\star) + d^2(x^A, x^\Delta) - 2V(\mathcal{X}_t, x^\star)d(x^A, x^\Delta)\cos(\theta) \leq \lambda^2 V^2(\mathcal{X}_t, x^\star) \tag{40}$$

By eq. (36), eq. (40) becomes:

$$V^2(\mathcal{X}_t, x^\star) + \lambda_2^2 V^2(\mathcal{X}_t, x^\star)\cos^2(\theta) - 2V^2(\mathcal{X}_t, x^\star)\lambda_2\cos^2(\theta) \leq \lambda^2 V^2(\mathcal{X}_t, x^\star), \text{ for } t = t^\star + 1. \quad (41)$$

And finally,

$$1 + \cos^2(\theta)(\lambda_2^2 - 2\lambda_2) < \lambda^2. \quad (42)$$

Ineq. (42) is satisfied for any $\lambda_2 \in (0, 2)$ and $\lambda \in (0, 1)$, thus, $V(\mathcal{X}_{t+1}, x^\star) < \lambda V(\mathcal{X}_t, x^\star)$, for $t = t^\star + 1$. □

Theorem IV.1. *Consider the system* (2)*,* (3)*. Then, by repeatably, invoking the remote estimation and performing first rotational motion and then translational motion, the robot converges to the target set* $\mathcal{G}$ *eq.* (24)*.*

Proof. By Lemma 1, after the remote estimation technique is invoked, the shift is orientation defined by problem (30a), guarantees that the translational motion is allowed in the next step, i.e.: $\mathcal{X}(t^\star + 1) \subseteq \mathcal{M}$. Furthermore, by Proposition 2, there is guarantee that $V(\mathcal{X}_{t+1}, x^\star) \leq \lambda V(\mathcal{X}_t, x^\star)$ for $t = t^\star + 1$, namely the distance from the target position is decreasing. The robot repeats the following process when constantly invoking the remote estimation technique, namely (i) performing a rotational motion and (ii) proceeds after one time step to the translational motion. Consequently, for any initial condition $\mathcal{X}_0$ such that $V(\mathcal{X}_0, x^*) \leq c$, it follows that $V(\mathcal{X}_t) \leq \lambda^{\lfloor \frac{N}{2} \rfloor} V(\mathcal{X}_0, x^*) \leq \lambda^{\lfloor \frac{N}{2} \rfloor} c$, thus, the robot converges to $\mathcal{G}$ as defined in eq. (24), in at most $N^* \geq 2\lfloor \frac{\log \epsilon - \log c}{\log \lambda} \rfloor$ steps. □

Following this simple technique, i.e., initially rotating to fix the heading error and then proceed to translational motion, it is apparent that the convergence of the proposed technique is guaranteed in the case of the robot constantly invokes the remote estimation. As stated before, this computationally intensive method heavily affects the mission duration of the robot. Thus, while relying on the theoretical guarantee, next we need to find an offloading strategy to find a balance between accuracy and mission duration, whilst the dynamic conditions of the network and remote computation resources, are also considered.

V. Computational Offloading Decision Mechanism

In this section, the decision-making mechanism behind the offloading of the computationally intensive tasks of our framework is presented. This mechanism is based on a utility function decision-making procedure that quantitatively ranks the current conditions, in terms of their associated resource metrics (i.e., availability of networking and computing resources at the edge and quality of robot's navigation) and dynamically assigns tasks to be executed on the edge infrastructure. To get a reliable estimation of these conditions, network and computing resource profiles are developed, as described in the following subsections.

A. Network Profiling

'q wsac dIn the following, mainly for demonstration purposes, we assume that the wireless access technique between the robot and the access point is based on IEEE 802.11g. In this network deployment, a common effect that occurs when a signal travels through a communication channel is its power level decreases as the distance increases. To estimate this propagation loss, the well-accepted Log-Distance Path Loss (LDPL) model is utilized [37]. The LDPL model applies to indoor environments with the presence of obstacles, having a propagation exponent that indicates whether the environment has more or fewer obstacles, impacting on the computed loss. The respective path-loss is calculated as follows:

$$PL(d)_{dB} = PL(d_0)_{dB} + 10n log_{10}(\frac{d}{d_0}),\ d \geq d_0, \quad (43)$$

where $PL(d_0)_{dB}$ is the path-loss at a reference distance $d_0 = 1m$, n is the path-loss exponent (PLE), which depends on the presence of obstacles in the environment. To set the upper bounds of the channel capacity we also leverage the signal-to-noise-ratio (SNR) metric,

$$SNR(d) = P_{dB} - PL(d)_{dB} - N_{dB}, \quad (44)$$

where P_{dB} is the incoming signal to the access point and N_{dB} is a Gaussian noise. Then, the channel capacity C can be calculated using the Shannon–Hartley theorem,

$$C(d) = B\ log_2(1 + SNR(d)), \quad (45)$$

where B is the available WLAN bandwidth (in Hz), giving in this way an estimation of the tightest upper bound on the information rate of data (in bits per second) that can be communicated at an arbitrarily low error rate using SNR. Having this bound available, an estimation of the task transmission duration (in seconds) can be calculated as follows:

$$e_{ul}(d) = \frac{8\ m}{C(d)}, \quad (46)$$

where m is the size of the offloaded data in bytes.

B. Edge Computing Resources Profiling

Regarding the profiling of the computing resources at the edge, we assume that the allocation of the resources to the localization service on the server is managed by the resource orchestrator of the infrastructure provider. Hence, we can only estimate the amount of allocated resources through measurements. To this end, we model the resource allocation strategy on the edge server as a linear dynamical system, subject to process and measurement uncertainty disturbances which follow the standard Kalman Filtering estimation approach [38], a computationally light prediction method. Briefly, using previous CPU utilization measurements, allows for acquiring a current estimation of the virtual CPU cores allocated to the container, $\hat{c}(t)$, dedicated the remote estimation technique. Subsequently, this allows for estimating the expected computation time (in seconds) of the offloaded

task to the edge server, $e_{comp}(t)$, by modeling it as a linear relationship of the available resources, specifically:

$$e_{comp}(t) = \alpha\ \hat{c}(t) + \beta. \tag{47}$$

The coefficients α, β are calculated using a least-squares fitting method on a set of pairs $(e_{comp}(t), \hat{c}(t))$ produced offline while experimenting with a dataset of pictures from the robot's camera. The interested reader may also refer to [7] for further information on this process, as it is based on this previous work.

C. Utility-based Offloading Strategy

After evaluating the current network and available edge computing resources, we apply the results in a utility function in order to quantify the trade-off between remote execution time and navigation performance. Naturally, this utility function incorporates a term, which assesses the quality of the navigation till the point of the offloading decision making, t, which in our case is expressed as a fraction (rational number) that is the quotient of the volume of the estimation set $\hat{\mathcal{X}}_t$, divided by the robot's remaining maximum distance of the estimation set towards the target position, $V(\hat{\mathcal{X}}_t, x^\star)$, as introduced in Section IV. The rationale behind this inclusion lies in the fact that the smaller the distance is between the robot and the target, the more precise the navigation has to be in order to approach it ϵ-close, thus the more preferable the offloading. Additionally, in Subsection IV-C, we identified a hard constraint for resorting to remote execution, expressed by the value of $\kappa(t)$ in eq. (33), which also has to be engaged in the offloading decision.

Therefore, taking the above into consideration, the utility function of edge computing processing for the robot can be defined as:

$$q_o(t) = c_1 \cdot \frac{Vol(\hat{\mathcal{X}}_t)}{V(\hat{\mathcal{X}}_t, x^\star)} - c_2 \cdot e_o(d,t) + \kappa(t) \cdot C, \tag{48}$$

where $e_o(d,t) = e_{ul}(d) + e_{comp}(t)$ denotes the total estimated duration of the remote execution. We should note here that, similarly to other studies [39], in the envisioned application, the duration of the transmission of the computation results is negligible, as the size of the computation results (plain text) is much smaller than the size of the input data (image or video). Thus, in our case, the total duration specifically involves the uplink transmission and the computation execution time. The $c_1, c_2 \geq 0, \in \mathbb{R}$ coefficients are carefully selected after thoroughly experimenting with different combinations in order to reflect the desired balance of mission duration and accuracy. $C \geq 0, \in \mathbb{R}$ represents a very large number.

By carefully examining eq. (48), one can notice that the system's utility increases as the navigation accuracy decreases, while decreases as the total offloading duration increases and is maximized when offloading is deemed necessary, i.e., $\kappa(t) = 1$. Leveraging the above, we define $O(t)$, a function that dictates whether the task is offloaded to the edge ($O(t) = 1$) or not ($O(t) = 0$), as follows:

$$O(t) = \begin{cases} 1, & \text{if } q_o(t) \geq J_{TH} \\ \\ 0, & \text{otherwise,} \end{cases} \tag{49}$$

where $J_{TH} \ll \mathcal{C}$ is a predefined threshold, the value of which, together with the values of c_1, c_2 and C, tunes the sensitivity of our mechanism in triggering the remote execution technique and depends on the mission's characteristics.

D. Convergence of the Proposed Technique and Core Algorithm

To summarize the whole operation of the proposed CPS, as introduced in Section IV, the robot's motion, dictated by the control automaton of Fig. 2, relies at first on local techniques for pose estimation. On the one hand, the robot performs a translational motion using the respective controller computed by eq. (25) and (26), only when it is guaranteed that the distance towards the target position for the whole estimation set decreases. On the other hand, the robot performs a rotation using the controller computed by problem (30a), only when it is guaranteed that in the next step the translational motion is allowed. In both cases, an overapproximation of the estimation set is calculated, as explained in Subsection III-B2. The offloading strategy switches to the remote estimation according to the system's utility, given by the utility function (48). This utility function dictates the robot to invoke the remote estimation whenever at least one of the two conditions for convergence, eq. (29) and (32), is not satisfied, or when the communication conditions and/or the available computing resources result in a fast computation for the precise remote estimation technique. In this case, given the precise estimation, by Theorem IV.1, the distance between the robot and the target position decreases. Thus, using a combination of the local and remote estimation technique, the robot eventually converges to the target position.

Corollary 1. *Consider the system* (2)*,* (3) *and the utility-based offloading strategy defined in* (48)*. Then, the robot converges to the target set $\mathcal{G}$ eq.* (24)*.*

Proof. The proof of Corollary 1 follows the same reasoning as in Theorem IV.1, with the exception that the convergence speed is $\hat{\lambda} = \max_t \lambda'(t)$, as given by eq. (29) and that the rotation is invoked at worst every two time instants. □

Algorithm 1 is also presented in order to provide the reader with a summarized outline.

VI. Performance Evaluation

In this section, we provide a detailed numerical performance evaluation of the proposed technique, through modeling and simulation of various scenarios, illustrating the operation, features and benefits of our approach. Specifically, in Subsection VI-A, the detailed configuration of the experimental setup is described. In Subsection VI-B, we focus on the control sequence benchmarking. To this end,

Algorithm 1: The algorithm of the proposed technique.

Data: Initial estimation set $\mathcal{X}_0 \subset \mathbb{R}^3$
Result: The robot reaches ϵ-close to the target $x^\star$

function `Main(`$\mathcal{X}_0$`)`**:**
  while $V(\hat{\mathcal{X}}_t, x^\star) > \epsilon$ **do**
    `// Target not reached`
    if $x^{repr} \in \mathcal{M}$ **then**
      1: **compute** the translational control action $u_1(t)$: Problem (25) and eq. (26)
      2: **compute** an approximation of the reachable set: $\hat{\mathcal{Z}}_{t+1}$, Sec. III-B1
        `// Convergence Constraint`
      if *ineq.* (29) *is not satisfied* **then**
        go to 5
      end
      call *Calculate* $\hat{\mathcal{X}}_t$
    else
      3: **compute** the rotational control action $u_2(t)$: Problem (30a)
      4: **compute** an approximation of the reachable set: $\hat{\mathcal{Z}}_{t+1}$, Sec. III-B1
        `// Convergence Constraint`
      if *ineq.* (32) *is not satisfied* **then**
        go to 5
      end
      call *Calculate* $\hat{\mathcal{X}}_t$
    end
    5: **estimate** the execution time of the remote estimation algorithm: $e_{ul}(d)$, eq. (46)
    6: **estimate** the execution time of the remote estimation algorithm: $e_{comp}(t)$, eq. (47)
    7: **compute** the Utility-based Offloading Strategy: $O(t)$, eq. (48)
    if $O(t) == 1$ **then** `// Offloading`
      8: **invoke** precise remote estimation, Sec. (IV-C)
      9: **compute** the rotational control action $u_2(t)$: Problem (30a)
      call *Calculate* $\hat{\mathcal{X}}_t$
      10: **compute** the translational control action $u_1(t)$: Problem (25) and eq. (26)
      call *Calculate* $\hat{\mathcal{X}}_t$
    end
  end
  11: Stop `// Target reached`

function *Calculate* $\hat{\mathcal{X}}_t$**:**
  1: **wait until** the motion is performed and perform local estimation using measurements
  2: **compute** an approximation of the reachable set: $\hat{\mathcal{Z}}_{t+1}$, Sec. III-B1
  3: **compute** the estimation set $\hat{\mathcal{X}}_{t+1}$: Sec. III-B2
return

we identify two metrics, based on which we evaluate the performance of our mechanism, namely the *a) navigation accuracy* and *b) mission duration* and different offloading strategies are compared. In Subsection VI-C, the utility function is evaluated for different application scenarios, indicating the easily adapted applicability of the proposed technique. The benchmarking is conducted by simulating the motion of a mobile wheeled robot in Python. For the calculation of $\hat{\mathcal{X}}$, we utilize Flow*, described in [40] and [41]. Flow* is a software used for calculating reachable sets in hybrid automata, together with other dedicated software (e.g., CORA[1]) [42]. The simulation code, alongside any related dataset used in this section, is publicly available[2].

A. Experiment Setup

In order to have a realistic setting, for evaluating the described scenarios, we simulate a square, $20m \times 20m$ sized factory floor where five wireless access points are located. Four of them are placed in the corners of the floor and the last one is places in its center. As the simulated robot, an AlphaBot[3] is selected, thus, the local pose estimation is performed by using the robot's photoelectric sensors (encoders) attached to each wheel, which provide an estimation about the travelled distance and the shift in orientation. Moreover, the discretization time is set at $T = 0.1s$. The remaining key experiment parameters are listed in Table III, unless otherwise explicitly stated. Additionally, regarding the mission, we consider that the robot starts from a known position $\mathcal{X}_0$ and stops when it reaches ϵ-close to the target position $x^\star$; for illustration purposes and in order to be able to average the following results over multiple repetitions of the experiment, the starting and final position are kept always the same.

Parameter	Value
ϵ	$0.2\ m$
$\mathcal{D}$	$[0.1, 0.2]\ rad$
w_1	$[-0.2, 0.03]\ m$
w_2	$[-0.06, 0.04]\ rad$
$x^\star$	$(13, 14)$
$\mathcal{X}_0$	$(3, 2)$
$\mathcal{J}_{TH}$	10
$\mathcal{U}_1$	$[0, 5]\ m/s$
$\mathcal{U}_2$	$[0, 2\pi]\ rad/s$

TABLE III: Simulation parameters.

Regarding the networking settings, we assume a signal of power P_{dB} for the uplink, which is proportional to the distance between the robot and the access point it is connected to and which has a maximum value of $P_{dB}^{max} = 24dB$. Moreover, we fix $PL(d_0)$ at $-20dBm$, based on the work of [37], which presents an access point with the same characteristics of ours and the same reference distance. The path-loss exponent n is set equal to 3.5, a value typical for a factory floor setting [43]. The size of offloaded data, in MB, follows a uniform distribution with a mean value of 0.075 and variance equal to 0.25. The Gaussian Noise N_{dB} is set equal to $-114dB$ while

[1]https://swmath.org/software/25659
[2]https://github.com/Dspatharakis/Replan
[3]https://www.waveshare.com/wiki/AlphaBot

the bandwidth B allocated to the robot at any given time is set to $1MHz$.

Finally, regarding the edge computing resources, as mentioned in subsection V-B and similar to [7], a least-squares fitting method is used to calculate the coefficients $\alpha = -1.34$ and $\beta = 3.675$ for the computation of the remote estimation technique. Also, the allocated cores of the edge server $\hat{c}(t)$ are updated every $0.5s$, following a Normal Distribution with a mean value of 0.75 and variance equal to 0.5.

B. Control Sequence Benchmarking

In this scenario we compare the performance of the following three computational offloading schemes, in terms of navigation accuracy and mission duration: *i) exclusively local estimation*, where the robot never invokes the remote estimation algorithm, *ii) exclusively remote estimation*, where the robot constantly invokes the estimation algorithm, *iii) utility-based offloading decision*, where our proposed mechanism comes into operation. For each of these three cases, a set of 35 experiments is conducted and the results are averaged per time slot for better illustration.

Fig. 5 depicts the changes in the estimation set volume (per second), as the robot moves and offloads tasks. The estimation set is depicted in 2-D, as the robot's position (and not its orientation) is sufficient to satisfy the termination condition (eq. (24)). Alongside this, Fig. 6 shows how quickly the robot reaches ϵ-close to the target in each case. As expected, in case *i)*, where only local estimations are used, although the robot moves rather quickly as shown in Fig. 6a, it consistently fails to approach the target as the localization error is accumulated in each step. This is illustrated in Fig. 5a. There, the vast increase of the size of the estimation set, due to this accumulated error, indicates that the robot can not converge to the target position. It must be noted that, in the 35 repetitions of this setting, only once the robot managed to reach ϵ-close to the target, as shown in Table IV. On the other hand, case *ii)* showcases great precision in localization, which also reflects at the size of the volume of the estimation set, which is fixed at the minimum value (point) throughout the mission, as shown in Fig. 5b. However, this comes with a cost in mission duration which is significantly increased due to the robot uncritically invoking a time-consuming estimation technique, despite the networking conditions and available edge computing resources not being always favorable.

After the evaluation of the first two cases, it becomes evident that a balance between the investigated metrics, i.e., navigation accuracy and mission duration, is of paramount importance and our method, *iii*), manages to deliver one.

As illustrated in Fig. 5c, in the beginning, where the robot moves quickly using the local pose estimation, the estimation set increases over time. Then, the utility-based offloading strategy invokes the remote estimation technique three times, specifically at the $16th$, $25th$ and $29th$ second of the experiment. In these moments, a precise estimation of the robot's pose is provided, thus the estimation set is minimized to a point. After offloading, the robot moves again faster towards the target position, as long as the size of the estimation set allows it. In this setting, the average mission duration is kept to a low $33s$, which is only 10% longer in duration than the single successful try of case *i)*. We should note here that the coefficients of the utility function of eq. (48) were fixed to values that provided a balanced outcome between the evaluation metrics, in order to reflect a generic mission. In the next subsection, an evaluation on the different values and dynamics between these coefficients follows.

Offloading Scheme	Average mission duration (s)	Average offloading triggers	Success Rate
Exclusively Local	25	0	3%
Exclusively Remote	270	95	100%
Utility-based	33	3	100%
Time-triggered ($\mu = 5$)	95	26	100%
Time-triggered ($\mu = 10$)	58	11	100%
Time-triggered ($\mu = 15$)	48	7	100%
Time-triggered ($\mu = 35$)	35	3	100%
Time-triggered ($\mu = 50$)	38	3	95%

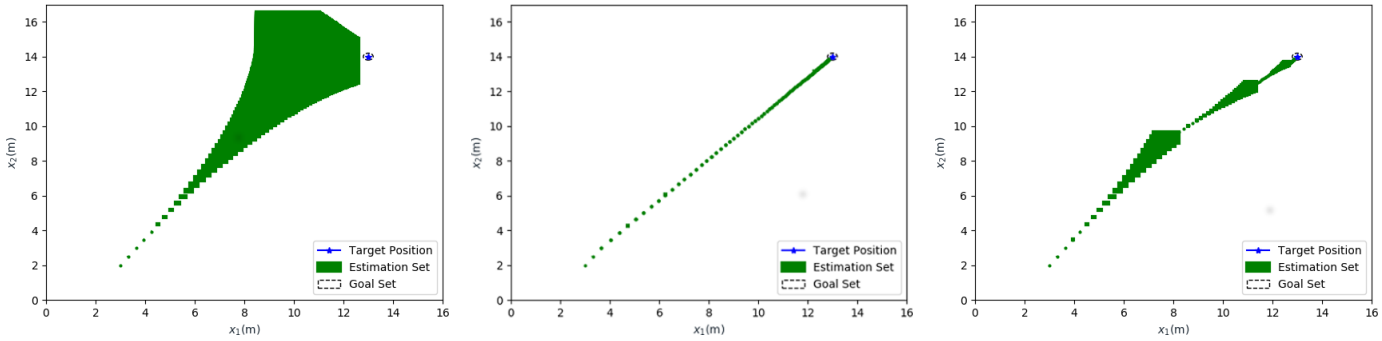
TABLE IV: Experiment metrics for each offloading scheme.

Finally, along with the three aforementioned offloading schemes we evaluate five different settings of a *time-triggered* remote estimation, a variation of case *iii)*, where the remote estimation is now triggered periodically every μ seconds. Although this time-triggered estimation provides, in some cases, similar results to our case, it has no convergence guarantees. This is elucidated in Table IV, where the averaged results for the set of 35 experiments, for all the offloading schemes are presented. As one can notice, our technique is superior to all the other offloading strategies when it comes to providing a guaranteed convergence to the navigation target, while keeping the mission duration low.

C. Offloading Strategy Benchmarking

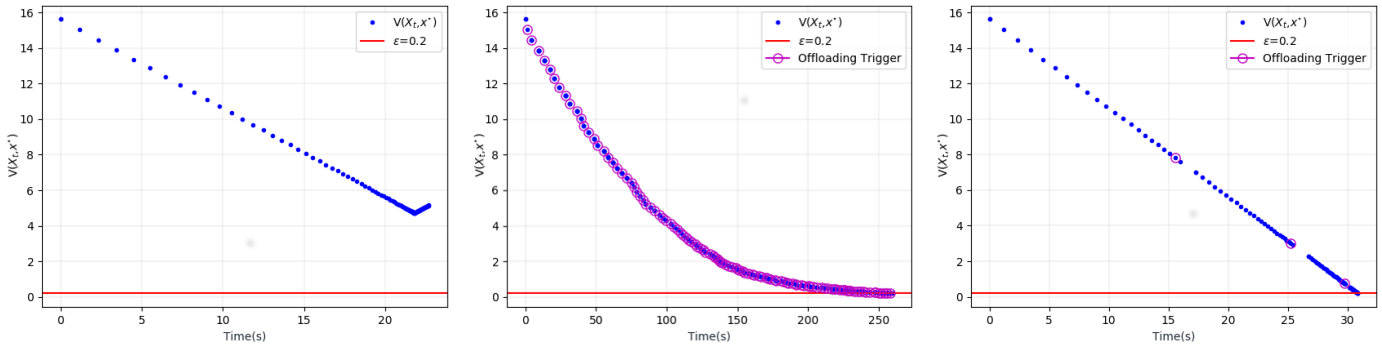
Having evaluated the overall usefulness of the proposed mechanism, in this section, we investigate the effect of the coefficients of the utility function, on navigation accuracy and mission duration. To this end, we tweak the coefficients of eq. (48) to steer the offloading strategy towards benefiting one of the two metrics, depending on the mission requirements. In this context, we identify three mission-scenarios, differing on which term of the utility function gets promoted and the results this has on the mission metrics: *i) time-critical mission*, (e.g., rescue robots), *ii) navigation-critical mission*, (e.g., autonomous museum tour guides) and *iii)* sparse communication mission (e.g., space robotics). The first case differs from the exclusively local execution one of the previous subsection, as we guarantee a 100% mission success rate, i.e., the robot always reaches the target.

For case *i)*, we tweak the coefficients of the utility function to benefit the mission duration. Consequently, c_2, which is associated with the total duration of the remote execution of the precise estimation technique is promoted, while c_1, which is associated with the quality of the navigation, is demoted. Hence, the time-consuming remote estimation is invoked either when the total duration of remote execution is really low, compared to the local estimations, or when the estimation set has grown vastly, as shown in Fig 7a. In detail, the remote estimation technique is invoked two times in this experiment,



(a) Exclusively Local. (b) Exclusively Remote. (c) Balanced Utility-based.

Fig. 5: Estimation set propagation for the three offloading schemes.



(a) Exclusively Local. (b) Exclusively Remote. (c) Balanced Utility-based.

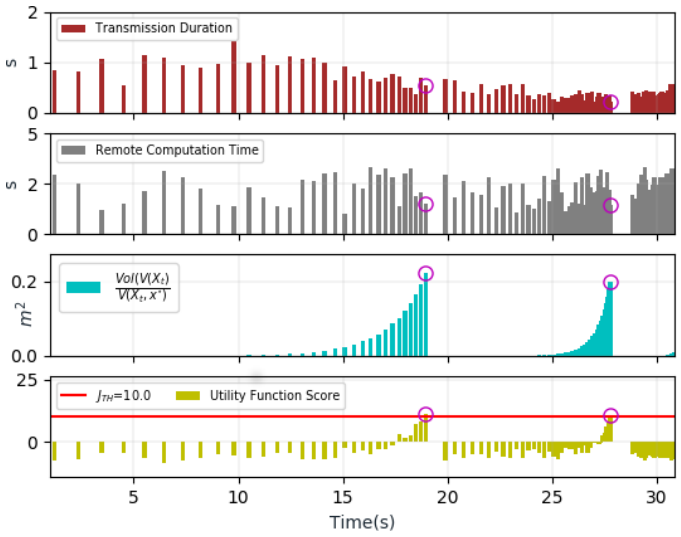
Fig. 6: Decrease of distance $V(\hat{\mathcal{X}}_t, x^\star)$ for the three offloading schemes.

when the volume of $\hat{\mathcal{X}}_t$ results in a great deterioration of the quality of the navigation, as shown in Fig. 7b and at the same time the total duration of remote execution is minimized. The average mission duration for this setting was $33s$.

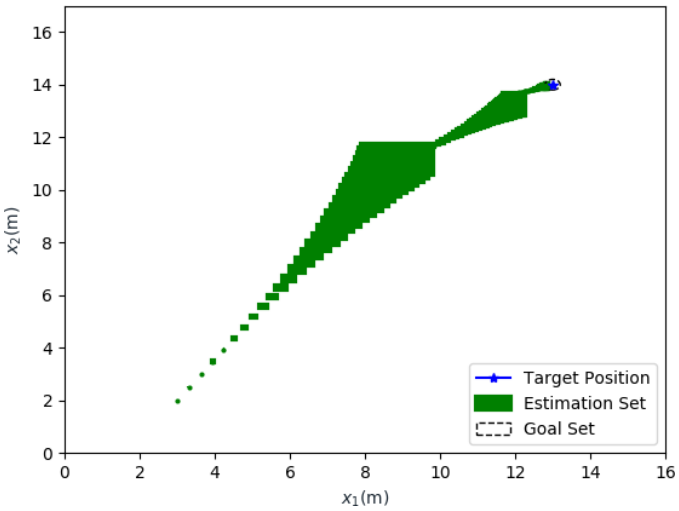
On the other hand, in case *ii)*, to address the need for a more fine-grained navigation, c_1 is promoted and c_2 is demoted, thus, the estimation set's volume is not allowed to grow uncontrollably. This results, as presented in Fig 8a, in invoking the remote estimation technique more frequently, i.e., 6 times during this experiment. In this case, the quality of navigation is more important than execution time; this is evident in the $1st$, $3rd$ and $6th$ offloading trigger, where the network conditions and available edge resources are not so preferable, but, still the robot chooses to offload in order to minimize the localization uncertainties. In addition, naturally, the volume of the estimation set, depicted in Fig. 8b, is relatively compared to the first case, resembling the case of exclusively offloading of Subsection VI-B. The average mission duration for this setting is 33% longer ($45s$) than case *i)*, as expected.

Finally, in case *iii)*, we minimize c_1, to simulate a scenario where the robot invokes the remote estimation algorithm only when it is an absolute necessity. This setting is a special case of case *i)* to indicate that the mission duration also depends on the increase of the estimation set. Therefore, as shown in Fig. 9a, in this case we notice only two offloading triggers, namely; the $1st$ due to the increase of the estimation set, at the $22nd$ second of the experiment and the $2nd$ due to the $\kappa(t)$ signal trigger at the $33rd$ second. As presented in Subsection IV-C, $\kappa(t)$ becomes equal to 1 either when the distance towards the target is not decreasing or the rotational controller fails to allow translational motion in the next time instant. As shown in the same figure, regardless of the current network and computing conditions, the robot seeks a more precise estimation to guarantee that the target is reached. A last note is that the average mission duration for this setting is 15% longer than time-critical experiment ($38s$). So, a conclusion that is drawn is that the bigger the estimation set, the more it deteriorates the quality of the navigation, under the proposed controller design.

To sum up, it is noticeable that promoting c_1 results in more precise yet time-consuming navigation, as the robot offloads more frequently. On the other hand, when promoting c_2, the robot offloads rarely, when absolutely necessary, resulting in quicker but less precise navigation. Moreover, in the special case, where c_1 is minimized, the robot offloads either when the estimation set grows vastly, or when the conditions that guarantee convergence are not satisfied. It should be noted that, while all three configurations manage to achieve a 100% success rate in reaching the target position, the first one is allowed to roam more freely, relying on its local navigation capabilities in favor of speed, the second one manages so in a more fine-grained trajectory while the third one invokes the remote estimation technique only when necessary. To sum up, the utility-based offloading decision framework manages to achieve the required balance among the various mission characteristics, making it suitable in the context of co-design

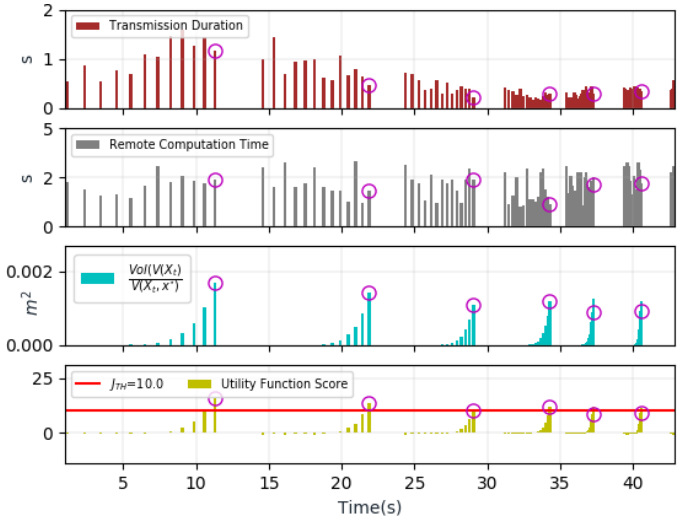


(a) Utility function breakdown.

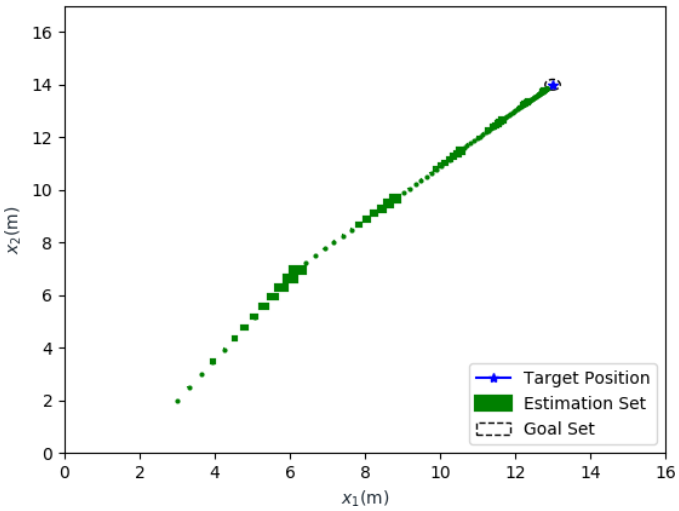


(b) Estimation set propagation.

Fig. 7: Time-critical mission.



(a) Utility function breakdown.



(b) Estimation set propagation.

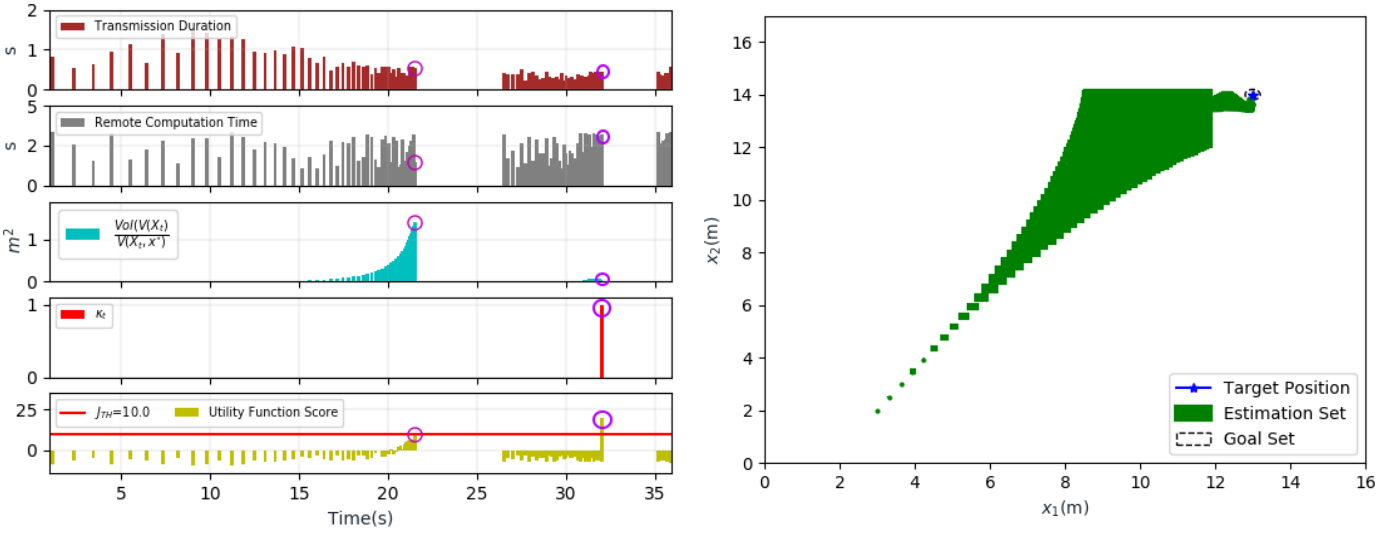
Fig. 8: Navigation-critical mission.

of 3C for a CPS.

VII. Conclusions and Future Work

In this paper, a novel resource-aware estimation and control framework for edge robotics, that jointly tackles the problem of convergence in trajectory navigation of a unicycle robot and the problem of efficiently using communication and computing resources, is presented. The conservative overapproximation techniques introduced alleviate additional computationally intensive tasks from the resource-constrained robot and provide a quick solution to the challenging problem of calculating the estimation set in the presence of modeling and measurement uncertainties. Moreover, we propose controllers that guarantee the robot reaches a target position after a finite number of steps. Finally, a utility-based offloading decision strategy is presented and thoroughly evaluated to highlight the need of finding a balance between two important metrics, namely navigation accuracy and mission duration. The performance evaluation of the proposed technique suggests that our solution outperforms other typically used offloading schemes and is easily adjustable to the needs of different application characteristics. More importantly, the proposed framework guarantees convergence to the target position independent of the various parameters chosen, in contrast to the periodic offloading schemes.

Our current and future research work focuses on further examining offloading capabilities in the context of 3C, especially the integration of planning algorithms and the adaptation of the offloading decisions based on safety guarantees for the robot's navigation. We also plan to investigate more sophisticated techniques to manipulate the unicycle dynamics [9], [23] and integrate them to the proposed set-based solution. Moreover, we aim to extend our work to cover the case of multiple robots moving in a common



(a) Utility function breakdown.

(b) Estimation set propagation.

Fig. 9: Sparse communication mission.

environment and using shared resources, thus requiring to adapt our framework to interacting agents in both the resource utilization problem and the trajectory tracking and path planning problem. Finally, deep learning techniques will be investigated, for calculating and dynamically adapting the utility function coefficients based on different application criteria.

Acknowledgement

This work was partially supported by the CHIST-ERA grant CHIST-ERA-18-SDCDN-003 and by Greek GSRT grant T11EPA4-00022.

References

[1] T. Taleb, I. Afolabi, and M. Bagaa, "Orchestrating 5G network slices to support industrial internet and to shape next-generation smart factories," *IEEE Network*, vol. 33, no. 4, pp. 146–154, 2019.

[2] D. Dechouniotis, N. Athanasopoulos, A. Leivadeas, N. Mitton, R. M. Jungers, and S. Papavassiliou, "Edge computing resource allocation for dynamic networks: The DRUID-NET vision and perspective," *Sensors*, vol. 20, no. 8, p. 2191, 2020.

[3] E. Uysal, O. Kaya, A. Ephremides, *et al.*, "Semantic Communications in Networked Systems," *arXiv preprint arXiv:2103.05391*, 2021.

[4] T. Abdelzaher, Y. Hao, K. Jayarajah, A. Misra, P. Skarin, S. Yao, D. Weerakoon, and K.-E. Årzén, "Five challenges in cloud-enabled intelligence and control," *ACM Transactions on Internet Technology (TOIT)*, vol. 20, no. 1, pp. 1–19, 2020.

[5] F. Saeik, M. Avgeris, D. Spatharakis, *et al.*, "Task offloading in Edge and Cloud Computing: A survey on mathematical, artificial intelligence and control theory solutions," *Computer Networks*, p. 108177, 2021.

[6] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Transactions on automation science and engineering*, vol. 12, no. 2, pp. 398–409, 2015.

[7] D. Spatharakis, M. Avgeris, N. Athanasopoulos, D. Dechouniotis, and S. Papavassiliou, "A Switching Offloading Mechanism for Path Planning and Localization in Robotic Applications," in *2020 International Conferences on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) and IEEE Congress on Cybermatics (Cybermatics)*, pp. 77–84, IEEE, 2020.

[8] S. L. Herbert, M. Chen, S. Han, S. Bansal, J. F. Fisac, and C. J. Tomlin, "Fastrack: A modular framework for fast and guaranteed safe motion planning," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 1517–1522, 2017.

[9] R. Olfati-Saber, "Near-identity diffeomorphisms and exponential/spl epsi/-tracking and/spl epsi/-stabilization of first-order nonholonomic se (2) vehicles," in *Proceedings of the 2002 american control conference (ieee cat. no. ch37301)*, vol. 6, pp. 4690–4695, IEEE, 2002.

[10] J.-C. Ryu and S. K. Agrawal, "Differential flatness-based robust control of mobile robots in the presence of slip," *The International Journal of Robotics Research*, vol. 30, no. 4, pp. 463–475, 2011.

[11] A. Bajcsy, S. Bansal, E. Bronstein, V. Tolani, and C. J. Tomlin, "An efficient reachability-based framework for provably safe autonomous navigation in unknown environments," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 1758–1765, IEEE, 2019.

[12] C. Liu, S. Lee, S. Varnhagen, and H. E. Tseng, "Path planning for autonomous vehicles using model predictive control," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 174–179, IEEE, 2017.

[13] K. Miller, C. Fan, and S. Mitra, "Planning in dynamic and partially unknown environments," in *In Proceedings of 7th IFAC Conference on Analysis and Design of Hybrid Systems (ADHS'21)*, 2021.

[14] L. Zhang, Z. Zhang, R. Siegwart, and J. J. Chung, "Optimized motion strategy for active target localization of mobile robots with time-varying connectivity," in *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pp. 185–187, IEEE, 2019.

[15] J. Xu, H. Cao, D. Li, K. Huang, C. Qian, L. Shangguan, and Z. Yang, "Edge assisted mobile semantic visual slam," in *IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pp. 1828–1837, IEEE, 2020.

[16] S. Chinchali, A. Sharma, J. Harrison, A. Elhafsi, D. Kang, E. Pergament, E. Cidon, S. Katti, and M. Pavone, "Network offloading policies for cloud robotics: a learning-based approach," *Autonomous Robots*, pp. 1–16, 2021.

[17] M. Nakanoya, S. Chinchali, A. Anemogiannis, A. Datta, S. Katti, and M. Pavone, "Task-relevant representation learning for networked robotic perception," *arXiv preprint arXiv:2011.03216*, 2020.

[18] A. W. Malik, A. U. Rahman, M. Ali, and M. M. Santos, "Symbiotic robotics network for efficient task offloading in smart industry," *IEEE Transactions on Industrial Informatics*, 2020.

[19] M. Avgeris, D. Spatharakis, N. Athanasopoulos, D. Dechouniotis, and S. Papavassiliou, "Single vision-based self-localization for autonomous robotic agents," in *2019 7th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, pp. 123–129, 2019.

[20] D. Pizarro, M. Mazo, E. Santiso, M. Marron, D. Jimenez, S. Cobreces, and C. Losada, "Localization of mobile robots using odometry and an external vision sensor," *Sensors*, vol. 10, no. 4, pp. 3655–3680, 2010.

[21] M. Lemmon, "Event-triggered feedback in control, estimation, and optimization," *Networked control systems*, pp. 293–358, 2010.

[22] S. Liu, D. E. Quevedo, and L. Xie, "Event-triggered distributed constrained consensus," *International Journal of Robust and Nonlinear Control*, vol. 27, no. 16, pp. 3043–3060, 2017.

[23] C. Ferrin, G. Droge, and R. Christensen, "Zero-error tracking for

autonomous vehicles through epsilon-trajectory generation," *IEEE Control Systems Letters*, vol. 5, no. 6, pp. 2084–2089, 2020.
[24] F. Blanchini and S. Miani, *Set-Theoretic Methods in Control*. Birkhäuser Basel, 1st ed., 2007.
[25] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.
[26] M. Ben-Ari and F. Mondada, *Robotic Motion and Odometry*, pp. 63–93. Cham: Springer International Publishing, 2018.
[27] X. Chen, E. Abraham, and S. Sankaranarayanan, "Taylor model flowpipe construction for non-linear hybrid systems," in *2012 IEEE 33rd Real-Time Systems Symposium*, pp. 183–192, IEEE, 2012.
[28] X. Chen, *Reachability analysis of non-linear hybrid systems using Taylor models*. PhD thesis, RWTH Aachen University, 2015.
[29] T. Dang, "Approximate reachability computation for polynomial systems," in *International Workshop on Hybrid Systems: Computation and Control*, pp. 138–152, Springer, 2006.
[30] T. Dreossi, T. Dang, and C. Piazza, "Parallelotope bundles for polynomial reachability," in *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control*, pp. 297–306, 2016.
[31] L. E. Dubins, "On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents," *American Journal of mathematics*, vol. 79, no. 3, pp. 497–516, 1957.
[32] L. Liebenwein, C. Baykal, I. Gilitschenski, S. Karaman, and D. Rus, "Sampling-based approximation algorithms for reachability analysis with provable guarantees," in *Proceedings of Robotics: Science and Systems*, (Pittsburgh, Pennsylvania), June 2018.
[33] A. Bais and R. Sablatnig, "Landmark based global self-localization of mobile soccer robots," in *Asian Conference on Computer Vision*, pp. 842–851, Springer, 2006.
[34] D. C. Yuen and B. A. MacDonald, "Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison," *IEEE Transactions on robotics*, vol. 21, no. 2, pp. 217–226, 2005.
[35] G. Vasiljević, D. Miklić, I. Draganjac, Z. Kovačić, and P. Lista, "High-accuracy vehicle localization for autonomous warehousing," *Robotics and Computer-Integrated Manufacturing*, vol. 42, pp. 1–16, 2016.
[36] W. Chen and T. Zhang, "An indoor mobile robot navigation technique using odometry and electronic compass," *International Journal of Advanced Robotic Systems*, vol. 14, no. 3, p. 1729881417711643, 2017.
[37] D. B. Faria *et al.*, "Modeling signal attenuation in ieee 802.11 wireless lans-vol. 1," *Computer Science Department, Stanford University*, vol. 1, 2005.
[38] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.
[39] K. Zhang, Y. Mao, S. Leng, S. Maharjan, and Y. Zhang, "Optimal delay constrained offloading for vehicular edge computing networks," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2017.
[40] X. Chen, E. Ábrahám, and S. Sankaranarayanan, "Flow*: An analyzer for non-linear hybrid systems," in *Computer Aided Verification* (N. Sharygina and H. Veith, eds.), (Berlin, Heidelberg), pp. 258–263, Springer Berlin Heidelberg, 2013.
[41] X. Chen and S. Sankaranarayanan, "Decomposed reachability analysis for nonlinear systems," in *2016 IEEE Real-Time Systems Symposium (RTSS)*, pp. 13–24, IEEE, 2016.
[42] M. Althoff, D. Grebenyuk, and N. Kochdumper, "Implementation of taylor models in cora 2018," in *Proc. of the 5th International Workshop on Applied Verification for Continuous and Hybrid Systems*, 2018.
[43] D. C. G. Valadares *et al.*, "802.11 g signal strength evaluation in an industrial environment," *Internet of Things*, vol. 9, p. 100163, 2020.

Dimitrios Spatharakis is currently a Ph.D. student and a research associate in the NETMODE Lab at the National Technical University of Athens (NTUA). He received a Diploma in Electrical & Computer Engineering (ECE) from NTUA, Greece, in 2018. His research interests focus on IoT, cyber-physical systems, edge computing and cloud computing. He has been involved as a researcher in several European and National R&D projects.

Marios Avgeris is currently a Ph.D student and a research associate in the NETMODE Lab at the National Technical University of Athens (NTUA). He received his Diploma in Electrical & Computer Engineering (ECE) from NTUA, Greece, in 2016. His research interests lie in the area of control theory, edge and cloud computing, IoT, semantic web technologies and network monitoring. He has been involved as a researcher in several European and National R&D projects.

Nikolaos Athanasopoulos Nikolaos Athanasopoulos is a Senior Lecturer at the School of Electronics, Electrical Engineering and Computer Science, Queen's University Belfast. He received a Diploma and a Ph.D in Electrical and Computer Engineering from the University of Patras, Greece, and has held postdoctoral researcher positions in TU/e, the Netherlands, and UCLouvain, Belgium. He has been awarded with an IKY (2006) and a Marie Curie (2012) Fellowship. His interests are in control theory with a focus on hybrid systems and set-based methods, with applications to robotics, edge computing and dynamic networks.

Dimitrios Dechouniotis is currently a senior research associate with NETMODE Lab of the National Technical University of Athens (NTUA). From 2007 to 2016, he was non-tenured Lecturer at the EE Dept. of Technical Educational Institute of Western Greece. He received his diploma in ECE from University of Patras in 2004, the MSc degree in Control Systems and Robotics from NTUA in 2009, and the Ph.D. degree in ECE from University of Patras in 2014. His research interests lie in the area of cloud computing, Internet of Things, trust management and control theory.

Symeon Papavassiliou (S'92-M'96-SM'11) Symeon Papavassiliou is currently a Professor in the School of Electrical and Computer Engineering at the National Technical University of Athens. From 1995 to 1999, he was a senior technical staff member at AT&T Laboratories, New Jersey. In August 1999 he joined the ECE Department at the New Jersey Institute of Technology, USA, where he was an Associate Professor until 2004. He has an established record of publications in his field of expertise, with more than 300 technical journal and conference published papers. His main research interests lie in the area of computer communication networks, with emphasis on the analysis, optimization, and performance evaluation of mobile and distributed systems, wireless networks, and complex systems. He received the Best Paper Award in IEEE INFOCOM 94, the AT&T Division Recognition and Achievement Award in 1997, the US National Science Foundation Career Award in 2003, the Best Paper Award in IEEE WCNC 2012, the Excellence in Research Grant in Greece in 2012, the Best Paper Awards in ADHOCNETS 2015, ICT 2016 and IEEE/IFIP WMNC 2019, as well as the 2019 IEEE ComSoc Technical Committee on Communications Systems Integration and Modeling best paper award (for his INFOCOM 2019 paper). He also served on the board of the Greek National Regulatory Authority on Telecommunications and Posts from 2006 to 2009.