

meSch: Multi-Agent Energy-Aware Scheduling for Task Persistence

Kaleb Ben Naveed¹, An Dang¹, Rahul Kumar¹, and Dimitra Panagou^{1,2}

Abstract—This paper develops a scheduling protocol for a team of autonomous robots that operate on long-term persistent tasks. The proposed framework, called **meSch**, accounts for the limited battery capacity of the robots and ensures that the robots return to charge their batteries one at a time at the single charging station. The protocol is applicable to general nonlinear robot models under certain assumptions, does not require robots to be deployed at different times, and can handle robots with different discharge rates. We further consider the case when the charging station is mobile and its state information is subject to uncertainty. The feasibility of the algorithm in terms of ensuring persistent charging is given under certain assumptions, while the efficacy of **meSch** is validated through simulation and hardware experiments. [\[Code\]^a](#)[\[Video\]^b](#)

I. INTRODUCTION

Autonomous robots are extensively used for persistent missions over long time horizons such as search and rescue operations [1], water body exploration [2], ocean current characterization [3], and gas leakage inspection [4].

Trajectory planning for persistent missions requires not only optimizing high-level objectives but also ensuring *task persistence*, realized in the sense that robots must be able to return for recharging as needed. When there is only one available charging station, the robots should coordinate their schedules to ensure the charging station’s exclusive use. Additionally, in persistent monitoring scenarios, maximizing the number of robots actively monitoring the environment is desirable, and ensuring that only one robot returns for recharging promotes this behavior. In this paper, we consider the above challenges, including the case where the charging station is mobile, and propose a solution framework **meSch**.

Recent work on task persistence has explored strategies for single-agent [5]–[9] as well as multi-agent scenarios [10]–[26]. In the realm of multi-agent task persistence, existing methods explore scenarios with both static [10]–[21] and mobile charging stations [21]–[27]. Most of the existing work on multi-agent task persistence with static charging stations assumes either a dedicated charging station for each agent [10]–[12], [21] or a single charging station that can cater to multiple agents simultaneously [13], [14]. Additionally, some studies address the scenarios where the number of charging stations available is fewer than the number of robots [15]–[17]. They do this by either altering the nominal paths to

visit charging stations along the way [15] or strategically placing the shared charging resources [16], or by imposing constraints to ensure that the number of returning robots matches the number of available charging stations [17]. In this paper, we explore a scenario where a team of robots exclusively shares a single charging station.

The closest previous works to this setup are [19], [20]. [19] ensured exclusive use of the charging station by deploying the robots at different times. Conversely, [20] ensured exclusive use by manipulating the minimum allowed SoC (State-of-Charge) of each robot’s battery using control barrier functions (CBF) [28]. While the CBF-based approach developed in [20] is computationally efficient, it is specifically tailored to the single-integrator model. In this paper, we propose an online scheduling method that ensures the exclusive use of the single charging station. The proposed method does not require robots to be deployed at different times or with different SoC capacities and applies to nonlinear robot models. Additionally, unlike [19], [20], we also address the scenario when the charging station is mobile.

Much of the literature addressing mobile charging stations considers that one of the robots in the team serves as a mobile charging station. Some approaches pre-plan the robots’ paths for the entire mission to determine appropriate rendezvous points where robots will meet for recharging during the mission [22], [23]. Others assume continuous communication between the robots to determine the rendezvous points online whenever one of the robots needs to recharge [24]. The other direction of work deploys separate mobile charging robot(s), and plans their paths to recharge the robots involved in the mission along their nominal paths [25]–[27].

In this paper, we consider a scenario where one of the robots in the persistent mission serves as a mobile charging station. Additionally, we address the case where robots have uncertain information about the mobile charging station’s position due to erroneous state estimation or external disturbances. With this in mind, we devise a method to robustly estimate rendezvous points online while accounting for this uncertainty. The paper is organized as follows: [Section II](#) formulates the problem statement, [Section IV](#) presents the proposed solution, and [Section V](#) discuss results.

II. PROBLEM FORMULATION

A. Notation

Let $\mathbb{Z}_0 = \{0, 1, 2, \dots\}$ and $\mathbb{Z}_+ = \{1, 2, 3, \dots\}$. Let $\mathbb{R}, \mathbb{R}_{\geq 0}, \mathbb{R}_{> 0}$ be the set of reals, non-negative reals, and positive reals respectively. Let \mathcal{S}_{++}^n denote set of symmetric positive-definite matrices in $\mathbb{R}^{n \times n}$. Let $\mathcal{N}(\mu, \Sigma)$ denote a normal

*The authors would like to acknowledge the support of the National Science Foundation (NSF) under grant no. 2223845 and grant no. 1942907.

¹Department of Robotics, University of Michigan, Ann Arbor, MI, 48109 USA. {kbnaveed@umich.edu}

²Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI, 48109 USA.

^aCodebase: <https://github.com/kalebbennaveed/meSch.git>

^bVideo: <https://youtu.be/wgH-KgNGJgw>

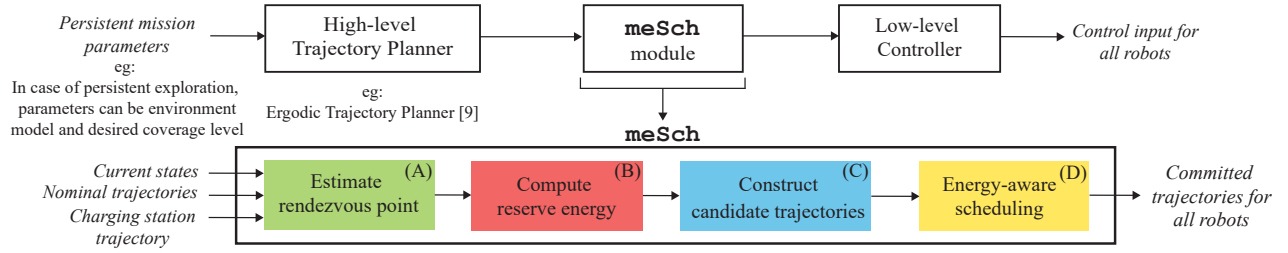


Fig. 1: **meSch**: Multi-Agent Energy-Aware Scheduling

distribution with mean μ and covariance $\Sigma \in \mathbb{S}_{++}^n$. The $Q \in \mathbb{S}_{++}^n$, norm of a vector $x \in \mathbb{R}^n$ is denoted $\|x\|_Q = \sqrt{x^T Q x}$.

B. System Description

Consider a multi-agent system, in which each robotic system $i \in \mathcal{R} = \{0, 1, \dots, N-1\}$, referred to as **rechargeable robot**, comprises the robot and battery discharge dynamics:

$$\dot{\chi}^i = \begin{bmatrix} \dot{x}^i \\ \dot{e}^i \end{bmatrix} = f^i(\chi^i, u^i) = \begin{bmatrix} f_r^i(x^i, u^i) \\ f_e^i(e^i) \end{bmatrix}, \quad (1)$$

where $N = |\mathcal{R}|$ is the cardinality of the set \mathcal{R} , $\chi^i = [x^iT, e^iT]^T \in \mathcal{Z}_r^i \subset \mathbb{R}^{n+1}$ is the i^{th} robotic system state consisting of the robot state $x^i \in \mathcal{X}_r^i \subset \mathbb{R}^n$ and its State-of-Charge (SoC) $e^i \in \mathbb{R}_{\geq 0}$. $u^i \in \mathcal{U}_r^i \subset \mathbb{R}^m$ is the control input, $f^i : \mathcal{Z}_r^i \times \mathcal{U}_r^i \rightarrow \mathbb{R}^{n+1}$ defines the continuous-time robotic system dynamics, $f_r^i : \mathcal{X}_r^i \times \mathcal{U}_r^i \rightarrow \mathbb{R}^n$ define robot dynamics and $f_e^i : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}$ define worst-case battery discharge dynamics. We also consider the continuous-time dynamics of the mobile charging station (referred to as **mobile charging robot**):

$$\dot{x}^c = f_c(x^c, u^c) + w(t), \quad w(t) \sim \mathcal{N}(0, W(t)), \quad (2a)$$

$$y^c = z(x^c) + v(t), \quad v(t) \sim \mathcal{N}(0, V(t)), \quad (2b)$$

where $x^c \in \mathcal{X}_c \subset \mathbb{R}^c$ is the charging station state, $u^c \in \mathcal{U}_c \subset \mathbb{R}^s$ is the charging station control input, $f_c : \mathcal{X}_c \times \mathcal{U}_c \rightarrow \mathbb{R}^c$ defines the continuous-time system dynamics for the mobile charging, $w(t)$ is the time-varying process noise with zero mean and known variance $W(t) \in \mathbb{R}_{\geq 0}$, $y \in \mathbb{R}^c$ is the measurement, $z : \mathbb{R}^c \rightarrow \mathbb{R}^c$ is the observation model, and $v(t)$ is the time-varying measurement noise with zero mean, and known covariance $V(t)$.

C. Problem Statement

Consider a team of $N+1$ robots performing a persistent mission. Among them, N robots are the rechargeable robots, require recharging, while 1 robot serves as the mobile charging robot that doesn't require recharging.^c The rechargeable robots model the mobile charging robot using (2).

We assume the existence of a high-level trajectory planner that designs the nominal trajectories for all $N+1$ robots based on the high-level mission objectives. The objectives for the rechargeable robots are as follows:

^cThis could represent a ground vehicle with a battery lasting a few hours. This assumption has been also made in previous works [25], [27].

1) Track the nominal trajectories and deviate only when returning for recharging.

2) Ensure mutually exclusive use of the mobile charging robot, which strictly follows its nominal trajectory.

Now we formulate the problem mathematically. We define \mathcal{T}^i as the set of times i^{th} robot returns to the charging station:

$$\mathcal{T}^i = \{t_0^i, t_1^i, \dots, t_m^i, \dots\}, \forall i \in \mathcal{R}, \forall m \in \mathbb{Z}_0 \quad (3)$$

where t_m^i represents the m^{th} return time of the i^{th} rechargeable robot. Let $\mathcal{T} = \cup_{i \in \mathcal{R}} \mathcal{T}^i$ be the union of return times for all robots. We now define two conditions that must hold for all times $t \in [t_0, \infty)$ to achieve the objectives stated above:

$$e^i(t) \geq e_{min}^i \quad \forall t \in [t_0, \infty), \forall i \in \mathcal{R} \quad (4a)$$

$$|t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta \quad \forall t_{m_1}^{i_1}, t_{m_2}^{i_2} \in \mathcal{T} \quad (4b)$$

Condition (4a), the **minimum SoC condition**, defines the required minimum battery SoC for all rechargeable robots. Condition (4b), the **minimum gap condition**, ensures a sufficient time gap between the returns of two robots to avoid charging conflicts. The term $T_\delta = T_{ch} + T_{bf}$ represents the charging duration and the buffer time needed for a robot to resume its mission before the next robot arrives. In summary, the problem statement is:

Problem 1. Given the nominal high level planner for all $N+1$ robots, design an algorithm to track the nominal trajectories for the N rechargeable robots while ensuring that conditions (4a), (4b) are met at all times.

Our goal in solving **Problem 1** is to develop an online algorithm for nonlinear robot models that does not assume a precomputed nominal trajectory for the entire mission but instead only requires a short horizon nominal trajectory at each decision iteration, allowing the high-level planner to adapt dynamically. The algorithm must handle varying discharge rates, account for erroneous state information about the mobile charging station available to rechargeable robots, and remain computationally efficient for scalability.

III. METHOD OVERVIEW, MOTIVATION & KEY IDEAS

We propose **meSch** as a solution to **Problem 1** within the persistent planning framework (Figure 1). As a low-level module, **meSch** ensures task persistence. The solution follows four steps, with the **meSch** module running every T_E seconds at discrete time steps $t_j = jT_E$, where $j \in \mathbb{Z}_0$:

- Compute the rendezvous point where the rechargeable robot will return for recharging.

- Determine the reserve energy at the rendezvous to account for uncertainty in the charging station's position.
- Construct a trajectory that follows the portion of the nominal trajectory and then reaches the rendezvous point. We refer to this as a *candidate trajectory*.
- Commit the new candidate trajectory if it satisfies the minimum energy condition (4a) and the minimum gap condition (4b). The result is the *committed trajectory*.

Before detailing each step, we first explain how **meSch** evaluates the satisfaction of conditions (4a) and (4b). This is one of our key contributions, and we explain it by first discussing its motivation and then describing its mechanism.

A. Method motivation

Consider N quadrotors sharing a mobile charging rover, as shown in Figure 2. To prevent charging conflicts, we propose a scheduling method based on two principles.

First, if multiple robots are predicted to arrive simultaneously, one is rescheduled to arrive earlier. Second, if robots naturally visit the charging station at different times due to varying discharge profiles, the algorithm checks that each robot has enough energy to continue its mission, ensuring that the minimum energy condition is never violated.

To implement this approach, we introduce two modules: **gware** and **eware**. The **gware** module maintains the minimum time gap between charging sessions by constructing *gap flags* and rescheduling robots to arrive early when conflicts arise, ensuring that the minimum gap condition (4b) is always satisfied. Once all gap constraints are met, the module **eware** checks whether each robot has enough energy to continue its mission, ensuring that the minimum energy condition (4a) is always met.

B. Construction of Gap flags

We begin by describing the construction of gap flags and their role in preventing charging conflicts. At each iteration of **meSch**, rechargeable robots are sorted by their remaining flight time into the ordered set $\mathcal{R}' = \{0', 1', \dots, N' - 1\}$, where $0'$ has the least flight time. For each robot $k \in \mathcal{R}' \setminus \{0'\}$, a gap flag is constructed relative to $0'$ as:

$$G^k = T_{F,j}^k > (T_R + T_E + kT_\delta), \quad (5)$$

where $T_{F,j}^k$ is k^{th} robot remaining flight time at time t_j , T_R is the time to reach the charging station, T_E is the decision interval, and T_δ includes the charging duration and the buffer time required to resume the mission. These flags enforce a minimum gap of kT_δ between robot $0'$ and robot k in \mathcal{R}' . For example, the minimum gap between the first and third robots is $2T_\delta$. If any gap flag is not satisfied, the robot with the least remaining flight time, i.e., $0'$, is rescheduled for recharging. The satisfaction of the gap flag condition guarantees that there will be at least T_δ between successive charging sessions.

IV. MULTI-AGENT ENERGY-AWARE SCHEDULING

In this section, we present the full proposed solution to Problem 1, using N rechargeable quadrotors and one mobile

Symbol	Definition
Indices	
i	Rechargeable robot index
j	meSch iteration index
k	Rechargeable robot index in sorted list
Constant shared time horizons	
T_δ	$T_\delta = T_{ch} + T_{bf}$ Charging + Buffer time
T_N	Nominal trajectory horizon of the rechargeable robot available at time t_j
T_R	Charging robot nominal trajectory horizon available at t_j / Time taken by the rechargeable robot to reach the charging station
T_E	Time interval between j and $j + 1$ iteration
Dynamic time horizons for robot i computed at t_j	
$T_{L,j}^i$	Worst-case landing time
$T_{C,j}^i$	Candidate trajectory ($T_{C,j}^i = T_R - T_{L,j}^i$)
$T_{B,j}^i$	back-to-base trajectory ($T_{B,j}^i = T_{C,j}^i - T_N$)
$T_{F,j}^k$	Remaining battery time of the k^{th} robot in the sorted list at time t_j
Time points	
t_j	Start time of iteration j
$t_{j,N}$	$t_j + T_N$
$t_{j,C}^i$	$t_j + T_{C,j}^i$
$t_{j,R}$	$t_j + T_R$
t_m^i	m^{th} time i^{th} robot returns for recharging

TABLE I: Time and Index Notation at a glance

charging rover, without loss of generality. After establishing the construction of gap flags, we demonstrate how they are iteratively checked within the full solution scheme to ensure conditions (4a) and (4b) hold for all $t \in [0, \infty)$. We also discuss how the proposed method accounts for the uncertainty in the position of the mobile charging robot. This solution is developed under a few key assumptions:

Assumption 1. At each iteration of **meSch**, the nominal trajectories of the rechargeable robots are known for T_N seconds, and the mobile charging robot's for T_R seconds.

We also define the notation for trajectories. Let $x^i([t_j, t_j + T_N]; t_j, x_j^i)$ represent the nominal trajectory for the i^{th} rechargeable robot at time t_j , starting from state x_j^i and defined over a time horizon of T_N seconds. We denote this as $x_j^{i,nom}$. The same notation applies to other trajectories. An overview of the notation is provided in Table I.

A. Estimating Rendezvous Point

At the j^{th} iteration of **meSch**, we estimate the mobile charging robot's position at $t_j + T_R$, i.e., $\hat{x}^c(t_{j,R})$, and place the rendezvous point d meters above it. The rechargeable robots will return to this point, as shown in Figure 2.

Given the current state estimate $\hat{x}^c(t_j)$ and its covariance $\Sigma^c(t_j)$ from the EKF, we use the EKF predict equations [29] to compute the mobile charging robot state estimate at $t_{j,R}$, i.e. $\hat{x}^c(t_{j,R})$ and $\Sigma^c(t_{j,R})$. The rendezvous point $x_j^{rp} \in \mathbb{R}^n$ is then computed as follows:

$$x_j^{rp} = \begin{bmatrix} \Psi(\hat{x}^c(t_{j,R})) \\ \mathbf{0}_{n-2} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_2 \\ d \\ \mathbf{0}_{n-3} \end{bmatrix} \quad (6)$$

where $\Psi : \mathbb{R}^c \rightarrow \mathbb{R}^2$ is a mapping that returns the 2-D position coordinates, $d \in \mathbb{R}_{>0}$ is added to the z-dim of the

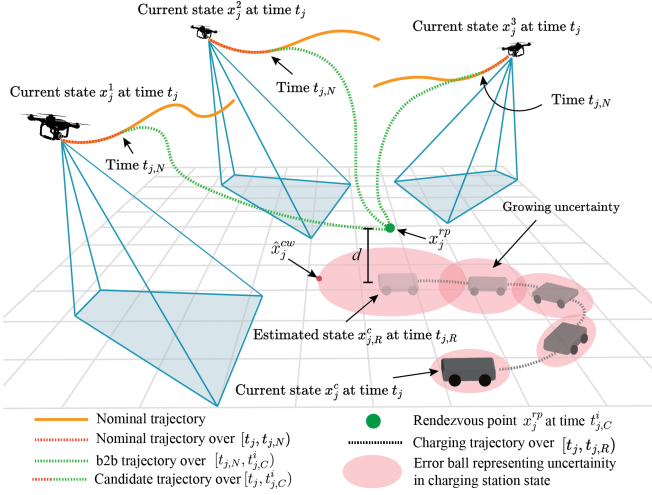


Fig. 2: This figure illustrates the generation of candidate trajectories at time t_j . All the candidate trajectories terminate at the rendezvous point x_j^{rp} at time $t_{j,C}^i$.

state, and n is the rechargeable robot state dimension (1). The rendezvous point corresponds to the hover reference state x_j^{rp} for the rechargeable robot, positioned d meters above the predicted position of the mobile charging robot.

B. Reserve Energy for Uncertainty-Aware Landing

Along with the rendezvous point x_j^{rp} , we also compute the remaining energy the robot must have at the rendezvous point to account for uncertainty in the mobile charging robot's position for landing. This corresponds to the energy cost of going from rendezvous point x_j^{rp} to the furthest state \hat{x}_j^{cw} within the 95% confidence interval covariance ellipse. Now, we compute the furthest point on the boundary of the 95% confidence ellipse as follows:

$$\hat{x}_j^{cw} = \hat{x}^c(t_{j,R}) + q_{max} \sqrt{\chi_{c,0.95}^2 \lambda_{max}} \quad (7)$$

where $\lambda_{max} \in \mathbb{R}$ is the largest eigenvalue of the covariance matrix $\Sigma^c(t_{j,R})$, $q_{max} \in \mathbb{R}^c$ is the eigenvector corresponding to λ_{max} , and $\chi_{c,0.95}^2$ corresponds to the value from the chi-squared distribution with c degrees of freedom in the 95% confidence interval. To compute the reserve energy, we formulate the following problem $\forall i \in \mathcal{R}$:

$$\min_{\chi^i(t), u^i(t), t_f^i} t_f^i \quad (8a)$$

$$\text{s.t. } \chi^i(t_0^i) = \chi_{rp}^i \quad (8b)$$

$$\dot{\chi} = f_r^i(\chi^i, u^i) \quad (8c)$$

$$x^i(t_f^i) = \hat{x}_j^{cw} \quad (8d)$$

where $\chi_{rp}^i = [[x_j^{rp}]^T, e_0^i]^T$ is the initial system state comprising of $x_j^{rp} \in \mathbb{R}^n$ and the energy $e_0 \in \mathbb{R}_{>0}$. The reserve energy $e_j^{i,res}$ and landing time $T_{L,j}^i$ are computed as follows:

$$e_j^{i,res} = e^i(t_f^i) - e^i(t_0^i) \quad \forall i \in \mathcal{R} \quad (9a)$$

$$T_{L,j}^i = t_f^i - t_0^i \quad \forall i \in \mathcal{R} \quad (9b)$$

Algorithm 1: The meSch Algorithm

```

1 function meSch( $x_j^{i,can}, x_{j-1}^{i,com}, e_j^{i,res}$ )
2   GapViolation = gware( $x_j^{i,can}, x_{j-1}^{i,com}$ )
3   if GapViolation == 1 then
4     return
5   eware( $x_j^{i,can}, x_{j-1}^{i,com}, e_j^{i,res}$ )

```

C. Construction of Candidate Trajectories

Now, we generate the candidate trajectories for all rechargeable robots to reach the rendezvous point x_j^{rp} from the current state $x^i(t_j)$ within $T_{C,j}^i = T_R - T_{L,j}^i$ s.

Given nominal trajectories $x_j^{i,nom} \forall i \in \mathcal{R}$, we construct a candidate trajectory that tracks a portion of the nominal trajectory for T_N s and then reaches the rendezvous point x_j^{rp} within $T_{B,j}^i = T_{C,j}^i - T_N$ s. For the i^{th} rechargeable robot, the candidate trajectory is constructed by concatenating the nominal trajectory with a **back-to-base (b2b) trajectory**. Let the i^{th} rechargeable robot state at time t_j be $x_j^i \in \mathcal{X}$ and the system state at t_j be $\chi_j^i \in \mathcal{Z}_r^i$. We construct a b2b trajectory $x_j^{i,b2b}$ defined over interval $[t_{j,N}, t_{j,C}^i]$ by solving:

$$\min_{x^i(t), u^i(t)} \int_{t_{j,N}}^{t_{j,C}^i} \|x^i(t) - x_j^{rp}\|_{\mathbf{Q}}^2 + \|u^i(t)\|_{\mathbf{R}}^2 dt \quad (10a)$$

$$\text{s.t. } x^i(t_{j,N}) = x_j^{i,nom}(t_{j,N}) \quad (10b)$$

$$\dot{x}^i = f_r^i(x^i, u^i) \quad (10c)$$

$$x^i(t_{j,C}^i) = x_j^{rp} \quad (10d)$$

where $\mathbf{Q} \in \mathbb{S}_{++}^n$ and $\mathbf{R} \in \mathbb{S}_{++}^m$ weights state cost and control cost respectively. Once b2b trajectory $x_j^{i,b2b}$ is generated, we numerically construct the system candidate trajectory

$$\chi_j^{i,can} = \begin{cases} x_j^{i,can}(t), & t \in [t_j, t_{j,C}^i] \\ e_j^{i,can}(t), & t \in [t_j, t_{j,C}^i] \end{cases} \quad (11)$$

over a time interval $[t_j, t_{j,C}^i]$ by solving the initial value problem for each rechargeable robot system, i.e.

$$\dot{\chi}^i = f(\chi^i, u^i(t)), \quad (12a)$$

$$\chi^i(t_j) = \chi_j^i \quad (12b)$$

$$u^i(t) = \begin{cases} \pi_r^i(\chi^i, x_j^{i,nom}(t)), & t \in [t_j, t_{j,N}] \\ \pi_r^i(\chi, x_j^{i,b2b}(t)), & t \in [t_{j,N}, t_{j,C}^i] \end{cases} \quad (12c)$$

where $\pi_r^i : \mathcal{Z}_r^i \times \mathcal{X}_r^i \rightarrow \mathcal{U}_r^i$ is a control policy to track the portion of the nominal trajectory and the b2b trajectory. Figure 2 shows the candidate trajectory generation process with 3 rechargeable robots and 1 mobile charging robot.

D. Energy-aware Scheduling

Given the candidate trajectory and the reserve energy for each rechargeable robot $i \in \mathcal{R}$, we check if the minimum SoC condition (4a) and the minimum gap condition (4b) are satisfied throughout the candidate trajectory. The overall algorithm described in Algorithm 1 consists of the two sub-routines: **gware** (gap-aware) and **eware** (energy-aware).

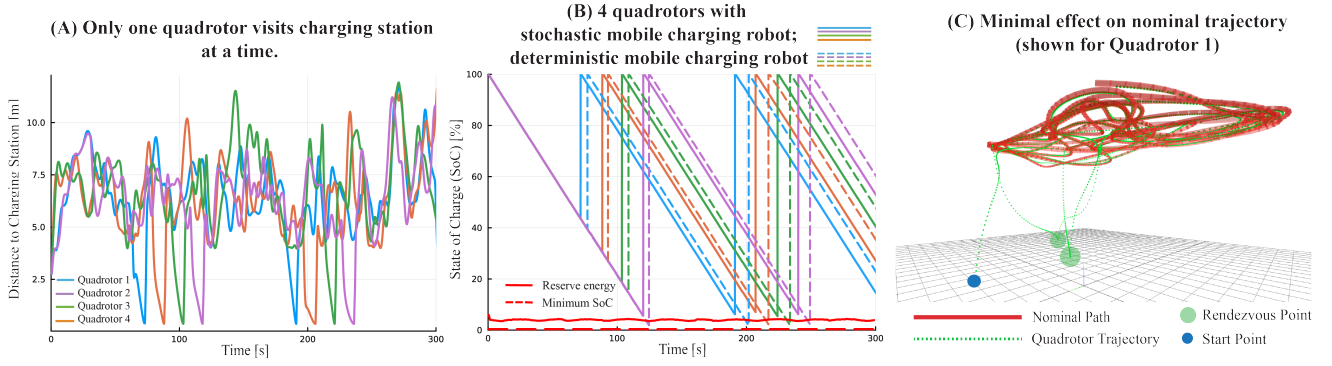


Fig. 3: Results from the Multi-Agent Energy-Aware Persistent Ergodic Search Case study

Algorithm 2: The **gware** algorithm

```

1 function gware( $x_j^{i,can}, x_{j-1}^{i,com}$ )
  // Sort  $x_j^{i,can}$  based on  $T_F^i$ 
2 for  $k \in \text{sorted list } \mathcal{R}' = \{1', 2', \dots, N' - 1\}$ : do
3    $G^k = (T_F^k - T_R - T_E) > k(T_\delta + T_\delta)$ 
4   if ( $G^k == 0$ )  $\wedge$  ( $k.\text{status} \neq \text{charging}$ ) then
5      $x_j^{0',com} \leftarrow x_{j-1}^{0',com}$ 
6      $x_j^{k,com} \leftarrow x_j^{k,can} \quad \forall k \in \mathcal{R}'$ 
7     Initiate landing for the  $0'^{th}$  robot at  $t_{j,C}^i$ 
8   return True

```

Algorithm 3: The **eware** algorithm

```

1 function eware( $x_j^{i,can}, x_{j-1}^{i,com}, e_j^{i,res}$ )
2 for  $i \in \{0, 1, \dots, N - 1\}$ : do
3   if  $e^i(t) \geq e_j^{i,res} \quad \forall t \in [t_j, t_{j,C}^i]$  then
4      $x_j^{i,com} \leftarrow x_j^{i,can}$ 
5   else
6      $x_j^{i,com} \leftarrow x_{j-1}^{i,com}$ 
7   Initiate landing for the  $i^{th}$  robot at  $t_{j,C}^i$ 

```

1) *Gap-aware (gware)*: **gware** described in Algorithm 2 checks if the rechargeable robots would continue to have the gap of T_δ seconds between their expected returns if the candidate trajectories were committed.

(Lines 2-4) Here the gap flags are constructed for each k^{th} robot that is not currently charging or returning, relative to the first robot in the sorted list (the 0^{th} robot)

$$G^k = T_{F,j}^k > (T_R + T_E + kT_\delta) \quad (13)$$

Satisfaction of the gap flag condition at the j^{th} iteration implies that rechargeable robots are estimated to have at least T_δ of the gap between their expected returns over the time interval $[t_j, t_{j,R}]$, i.e. $\forall t_{m_1}^{i_1}, t_{m_2}^{i_2} \in \mathcal{T}$:

$$T_{F,j}^k > (T_R + T_E + kT_\delta) \quad (14)$$

$$\implies |t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta \quad \forall t \in [t_j, t_{j,R}] \quad (15)$$

(Lines 5-7) If any gap flags are false, the committed trajectory of the first rechargeable robot in the sorted list remains

unchanged, and it returns to the charging station. Meanwhile, the candidate trajectories are committed for the subsequent rechargeable robots in the sorted list.

2) *Energy-aware (eware)*: If no gap flag violations occur, indicating that all rechargeable robots have sufficient gaps between their expected return for recharging, we proceed to check if each robot has adequate energy to continue the mission using **eware** described in Algorithm 3.

(Lines 3-6) We assess whether each rechargeable robot can reach the charging station without depleting its energy below the reserve level while following the candidate trajectory. We refer to this condition as the **Reserve SoC Condition**:

$$e^i(t) > e_j^{i,res} \quad \forall t \in [t_j, t_{j,C}^i] \quad (16)$$

If successful, the candidate trajectory replaces the current committed one. For the returning robot, a landing controller is assumed to exist:

Assumption 2. When the returning rechargeable robot reaches rendezvous point x_{rp}^c at $t_{j,C}^i$, there exists a landing controller $\pi_l^i : [t_{j,C}^i, t_{j,R}] \times \mathcal{X}_r^i \rightarrow \mathcal{U}$ that guides the rechargeable robot to the mobile charging robot.

Theorem 1. Suppose that at $j = 0$ the Gap flag condition (13) and the Reserve SoC condition (16) are satisfied. Therefore, the conditions (4a) and (4b) are satisfied for $[t_0, t_{0,R}^i]$. Following this, if solution exist for (8) and (10) and the committed trajectories $x_j^{i,com} = x^i([t_j, t_{j,C}^i]; t_j, x_j^i)$ are determined, $\forall i \in \mathcal{R}$, using the Algorithm 1, then (4a) and (4b) are satisfied $\forall t \in [t_j, t_{j-1,R}]$, $\forall j \in \mathbb{Z}_+$.

Proof. Please see Appendix A for detailed proof. \square

Given that the conditions (4a) and (4b) are met for the interval $[t_j, t_{j-1,R}] \quad \forall j \in \mathbb{Z}_+$, and $t_j < t_{j-1,N} < t_{j-1,R}$, the conditions (4a) and (4b) are satisfied over $[t_0, \infty)$.

V. RESULTS & DISCUSSION

In this section, we evaluate **meSch** through case studies and hardware experiments. We use quadrotors with 3D nonlinear dynamics from [31, Eq. (10)] as rechargeable robots and rovers with unicycle models as mobile charging robots. We assume instantaneous recharging ($T_{ch} = 0.0$ s) and a buffer time of $T_{bf} = 15.0$ s, with $T_N = 2.0$ s and $T_R = 18.0$ s, consistent across all experiments.

TABLE II: Comparison of baseline methods and proposed **meSch**

Method	Robot Model (Supports Nonlinear Dynamics)	Requires Different Deployment Times	Total Recharging Visits	Concurrent Charging Visits	Minimum Energy Violations	Scalability Analysis	Supports Mobile Charging Station
Baseline 1 [20]	3D Single integrator (No)	No	8	0	0	Not provided	No
Baseline 2 [19]	3D Quadrotor [30] (Yes)	Yes	8	0	0	Not provided	No
Baseline 3 [9]	3D Quadrotor [31] (Yes)	No	8	2	0	Not provided	No
Baseline 4 [meSch with only gware]	3D Quadrotor [31] (Yes)	No	4	0	4	$\mathcal{O}(N \log N)$	Yes
Proposed [meSch]	3D Quadrotor [31] (Yes)	No	8	0	0	$\mathcal{O}(N \log N)$	Yes

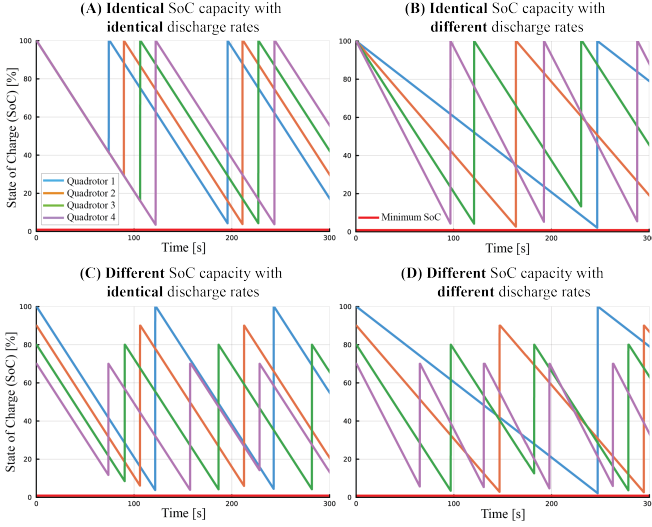


Fig. 4: These plots show results for the scenarios when four quadrotors have different SoC capacities and different discharge rates. The plots validate that quadrotors always maintain the minimum of $(T_c + T_\delta)$ gap while visiting the charging station.

To generate b2b trajectories, we solve (10) using MPC with the reduced linear quadrotor dynamics from [31]. We use an LQR controller for (8) and an LQG controller for landing. Trajectories are generated at 1.0 Hz and tracked at 50.0 Hz with zero-order hold, using the RK4 integration.

1) Multi-Agent Energy-Aware Persistent Ergodic Search: We evaluate **meSch** by simulating a scenario with four quadrotors and one rover exploring a 10×10 m domain. Nominal trajectories are collision-free ergodic trajectories with a horizon of $T_H = 30.0$ s. All quadrotors use discharge dynamics $\dot{e} = -0.667$. Coverage objectives are omitted for brevity. Figure 3 summarizes the results.

Figure 3 (A) shows that only one quadrotor is at the charging station at any time. Collision avoidance is ensured by tuning the T_δ parameter. Figure 3 (B) plots the SoC evolution for two cases: with a stochastic mobile charging robot and with a deterministic one. In the stochastic case, the quadrotors' SoC never drops below the reserve energy level and in the deterministic case, the SoC levels never drop below zero. Finally, Figure 3 (C) shows that the nominal ergodic trajectory (in red) is only affected when quadrotor 1 returns for recharging.

2) Comparison to the Baseline Methods: We compare **meSch** to baseline methods using seven metrics, as shown in Table II. For each method, four robots are used with the

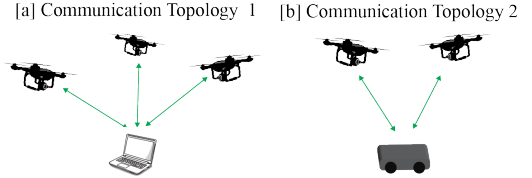


Fig. 5: The communication architecture of the system.

same discharge model, $\dot{e} = -0.667$. The total recharging visits are the same across all methods, except for Baseline 4, which focuses only on the timing of robot visits and does not account for the minimum energy requirements.

Compared to Baseline 1, **meSch** supports nonlinear dynamic models, making it more applicable to real-world robotic platforms, as demonstrated with 3D quadrotor dynamics [31]. Unlike Baseline 2, **meSch** effectively handles both identical and varying discharge rates and state-of-charge (SoC) capacities without requiring robots to be deployed at different times. Deploying robots at different times reduces the number of robots available for the mission at any given moment, limiting overall efficiency. By allowing all robots to be deployed simultaneously, **meSch** simplifies mission planning and increases adaptability to different discharge patterns, as shown in Figure 4 with four quadrotors. Compared to Baseline 3, **meSch** eliminates simultaneous charging station visits. In Baseline 3, four robots returned concurrently on two occasions, leading to a violation of (4b). While Baseline 4, which only includes the **gware** module from **meSch**, successfully avoids overlapping visits, it fails to enforce minimum energy constraints, resulting in a violation of (4a). Finally, none of the baseline methods support mobile charging stations, a limitation in the environments where fixed charging locations may be infeasible. By addressing this gap, **meSch** enhances mission endurance and scalability.

3) Computational efficiency and scalability: Distributing computation across the robot network improves the efficiency of the **meSch** module. The main overhead comes from generating candidate trajectories, with solving (10) and integrating the system's nonlinear dynamics taking 150 ms and 30 ms on average, respectively. We employ the communication architecture(s) shown in Figure 5.

To support real-time applications, each rechargeable robot (e.g. Quad 1) generates candidate trajectories on board, which are transmitted to the central node (Base) for scheduling. The scheduling algorithm has time complexity $\mathcal{O}(N \log N)$, mainly due to the sorting function in line 2

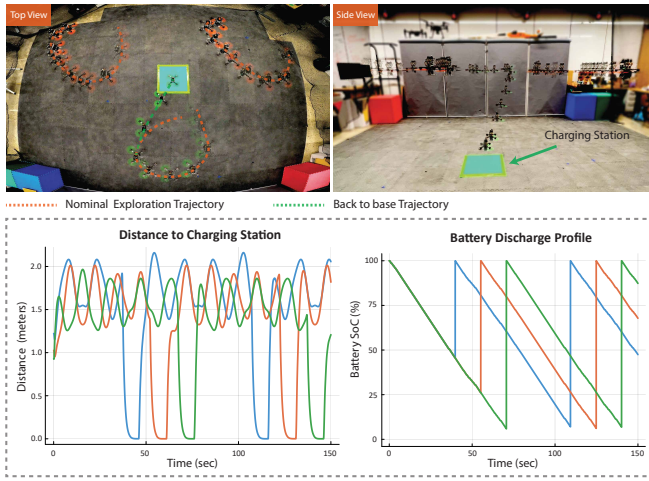


Fig. 6: The sequence of three quadrotor trajectories is shown, with one returning to the charging station. The plots display the distance to the station and the battery discharge profile over 150 seconds.

of Algorithm 2. Thus, the method scales with $\mathcal{O}(N \log N)$, where N is the number of rechargeable robots. To demonstrate scalability, we evaluate the method with 40 rechargeable quadrotors (shown in the attached video). In the static case, quadrotors return with $(3 \pm 1)\%$ battery SoC remaining. With the mobile charging robot, they return with $8 \pm 1.5\%$ SoC, accounting for reserve energy.

4) **Feasibility and Robustness of meSch:** If the Gap flag (13) and Reserve SoC (16) conditions hold in the first iteration, **meSch** guarantees the satisfaction of (4a) and (4b) for all times, assuming (8) and (10) are solvable as stated in Theorem 1. **meSch** is robust to rechargeable robot failures, as it reduces the number of gap flag conditions (13) to check, like removing an inequality constraint in optimization. However, it is not provably robust to new robots, as they might add potentially infeasible constraints. Additionally, **meSch** is vulnerable to central node failure, which could violate the minimum desired gap due to its centralized nature.

5) **Hardware Demonstration:** We validate **meSch** in two setups: (1) three quadrotors with a static charging station, and (2) two quadrotors with a rover as a mobile charging station. Each quadrotor uses an NVIDIA Orin NX for onboard computing, and the rover has a Raspberry Pi. The communication architecture is shown in Figure 5. Experiments were conducted in an indoor arena with 17 Vicon cameras for state estimation. Snapshots and plots are in Figure 6 and Figure 7. In both setups, quadrotors follow nominal Lissajous coverage trajectories, with only $T_N = 2.0$ s of the future nominal trajectory available at each iteration of **meSch**. Candidate trajectories are generated onboard the quadrotors and sent to the rover’s (or base) computer to verify gap flags and minimum state-of-charge (SoC) conditions. The rover continuously publishes its state and nominal trajectory to assist in trajectory generation. We also account for delays caused by computational overhead and ROS2 message latency in our implementation. The primary computational costs include candidate trajectory generation and forward propagation (T_1), gap flag construction and checking (T_2),

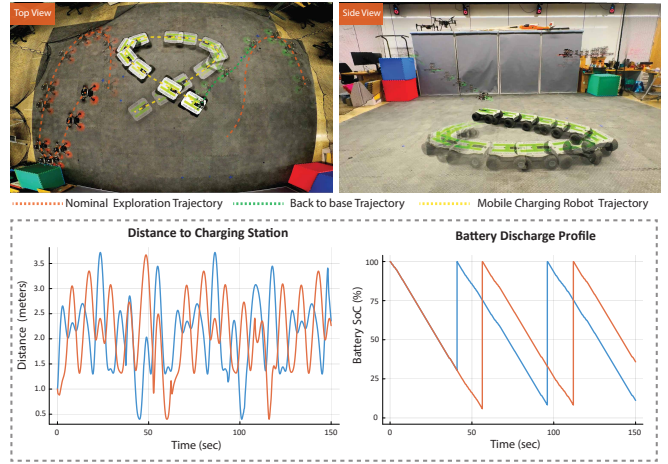


Fig. 7: The trajectories of two quadrotors and the rover over a short horizon are shown, with one quadrotor returning to the rover for recharging. The plots display the distance to the charging station and the battery discharge profile over the full 150-second mission.

and miscellaneous delays due to message latency in ROS2 (T_3). As long as $T_1 + T_2 + T_3 < T_E$, where T_E is the **meSch** decision interval, the algorithm ensures recursive feasibility as discussed in Theorem 1. In our experiments with three quadrotors, we observed a latency of 600 ± 150 ms, while T_E was set to 1.5 s.

We release all simulation and experimental code. **meSch** is available as a Julia module, functioning as a low-level filter for a planner. We also provide a Python-ROS2 wrapper for Julia, along with a Docker container for the experiments, and our in-house-built Orin-based DevQuad [32].

VI. CONCLUSION AND FUTURE WORK

In this work, we present **meSch**, a framework for managing the exclusive use of a mobile charging station by robots performing persistent tasks. Through analysis and case studies, we show that **meSch** efficiently schedules recharges for robots with nonlinear dynamics, varying battery capacities, and discharge rates, while accounting for uncertainty in the charging station’s position. Future work will enhance **meSch**’s robustness to central node failure via a distributed implementation, demonstrate its adaptability with continuously replanned trajectories, and refine candidate trajectory generation—especially b2b trajectories—by optimizing them for mission-specific goals like information maximization.

REFERENCES

- [1] Y. Li, Y. Gao, S. Yang, and Q. Quan, “Swarm robotics search and rescue: A bee-inspired swarm cooperation approach without information exchange,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 1127–1133.
- [2] S. Manjanna, A. Q. Li, R. N. Smith, I. Rekleitis, and G. Dudek, “Heterogeneous multi-robot system for exploration and strategic water sampling,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4873–4880.
- [3] R. E. Todd, “Export of middle atlantic bight shelf waters near cape hatteras from two years of underwater glider observations,” *Journal of Geophysical Research: Oceans*, vol. 125, no. 4, 2020.
- [4] S. Soldan, J. Welle, T. Barz, A. Kroll, and D. Schulz, “Towards autonomous robotic systems for remote gas leak detection and localization in industrial environments,” in *Field and Service Robotics: Results of the 8th Int. Conference*. Springer, 2014, pp. 233–247.

[5] W. Bentz and D. Panagou, "Energy-aware persistent coverage and intruder interception in 3d dynamic environments," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 4426–4433.

[6] G. Notomista, C. Pacchierotti, and P. R. Giordano, "Online robot trajectory optimization for persistent environmental monitoring," *IEEE Control Systems Letters*, vol. 6, pp. 1472–1477, 2022.

[7] K. Sundar and S. Rathinam, "Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots," *IEEE Transactions on Automation Science and Engineering*, vol. 11, no. 1, pp. 287–294, 2014.

[8] M. Wei and V. Isler, "Coverage path planning under the energy constraint," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 368–373.

[9] K. B. Naveed, D. Agrawal, C. Vermillion, and D. Panagou, "Eclares: Energy-aware clarity-driven ergodic search," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 14 326–14 332.

[10] G. Notomista, "Resilience and energy-awareness in constraint-driven-controlled multi-robot systems," in *2022 American Control Conference (ACC)*, 2022, pp. 3682–3687.

[11] G. Notomista, C. Pacchierotti, and P. R. Giordano, "Multi-robot persistent environmental monitoring based on constraint-driven execution of learned robot tasks," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 6853–6859.

[12] G. Notomista, S. F. Ruf, and M. Egerstedt, "Persistification of robotic tasks using control barrier functions," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 758–763, 2018.

[13] A. B. Asghar, S. Sundaram, and S. L. Smith, "Multi-robot persistent monitoring: Minimizing latency and number of robots with recharging constraints," 2023.

[14] M. Kenzin, I. Bychkov, and N. Maksimkin, "Coordinated recharging of heterogeneous mobile robot teams during continuous large scale missions," in *2020 7th International Conference on Control, Decision and Information Technologies (CoDIT)*, vol. 1, 2020, pp. 745–750.

[15] L. Liu and N. Michael, "Energy-aware aerial vehicle deployment via bipartite graph matching," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2014, pp. 189–194.

[16] B. Li, S. Patankar, B. Moridian, and N. Mahmoudian, "Planning large-scale search and rescue using team of uavs and charging stations," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, 2018, pp. 1–8.

[17] A. Sewald, C. J. Lerch, M. Chancán, A. M. Dollar, and I. Abraham, "Energy-aware ergodic search: Continuous exploration for multi-agent systems with battery constraints," 2024.

[18] J. Xu, J. Wang, and W. Chen, "An efficient recharging task planning method for multi-robot autonomous recharging problem," in *2019 IEEE International Conference on Robotics and Biomimetics (RO-BIO)*, 2019, pp. 1839–1844.

[19] W. Bentz, T. Hoang, E. Bayasgalan, and D. Panagou, "Complete 3-d dynamic coverage in energy-constrained multi-uav sensor networks," *Autonomous Robots*, vol. 42, pp. 825–851, 2018.

[20] H. Fouad and G. Beltrame, "Energy autonomy for robot systems with constrained resources," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3675–3693, 2022.

[21] T. Gao and S. Bhattacharya, "Multirobot charging strategies: A game-theoretic approach," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2823–2830, 2019.

[22] N. Karapetyan, A. B. Asghar, A. Bhaskar, G. Shi, D. Manocha, and P. Tokekar, "Ag-cvg: Coverage planning with a mobile recharging ugv and an energy-constrained uav," 2023.

[23] N. Kingry, Y.-C. Liu, M. Martinez, B. Simon, Y. Bang, and R. Dai, "Mission planning for a multi-robot team with a solar-powered charging station," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 5233–5238.

[24] T. X. Lin, E. Yel, and N. Bezzo, "Energy-aware persistent control of heterogeneous robotic systems," in *2018 Annual American Control Conference (ACC)*, 2018, pp. 2782–2787.

[25] N. Mathew, S. L. Smith, and S. L. Waslander, "Multirobot rendezvous planning for recharging in persistent tasks," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 128–142, 2015.

[26] A. Couture-Beil and R. T. Vaughan, "Adaptive mobile charging stations for multi-robot systems," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, pp. 1363–1368.

[27] X. Lin, Y. Yazıcıoğlu, and D. Aksaray, "Robust planning for persistent surveillance with energy-constrained uavs and mobile charging

stations," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4157–4164, 2022.

- [28] A. D. Ames, X. Xu, J. W. Grizzle, and P. Tabuada, "Control barrier function based quadratic programs for safety critical systems," *IEEE Trans. on Automatic Control*, vol. 62, no. 8, pp. 3861–3876, 2017.
- [29] A. Gelb *et al.*, *Applied optimal estimation*. MIT press, 1974.
- [30] R. Beard, "Quadrotor dynamics and control rev 0.1," 2008.
- [31] B. E. Jackson, K. Tracy, and Z. Manchester, "Planning with attitude," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5658–5664, 2021.
- [32] D. R. Agrawal, R. Chen, and D. Panagou, "gatekeeper: Online safety verification and control for nonlinear systems in dynamic environments," *IEEE Transactions on Robotics*, vol. 40, pp. 4358–4375, 2024.

APPENDIX

A. Proof of Theorem 1

Proof. The proof, inspired by [32, Thm. 1], uses induction.

Base Case: At the time t_1 and iteration $j = 1$, since both Gap flag condition (13) and the Reserve SoC condition (16) are true, the candidate trajectories are committed for all rechargeable robots i.e. $\forall i \in \mathcal{R}$ and $\forall t_{m_1}^{i1}, t_{m_2}^{i2} \in \mathcal{T}$

$$\begin{aligned} x_1^{i,com}(t) &\leftarrow x_1^{i,can}(t) \quad \forall t \in [t_1, t_{1,C}^i) \\ &\Rightarrow \begin{cases} T_{F,1}^k > (T_R + T_E + kT_\delta) & \forall k \in \mathcal{R}' \\ e^i(t) > e_1^{res} & \forall t \in [t_1, t_{1,C}^i) \end{cases} \\ &\Rightarrow \begin{cases} |t_{m_1}^{i1} - t_{m_2}^{i2}| > T_\delta & \forall t \in [t_1, t_{1,R}) \\ e^i(t) > e_{min}^i & \forall t \in [t_1, t_{1,R}) \end{cases} \end{aligned}$$

Since $t_{1,R} > t_{0,R} > t_{0,C}^i \forall i \in \mathcal{R}$, the claim holds.

Induction Step: Suppose the claim is true for some $j \in \mathbb{Z}_+$. We show that the claim is true for $j + 1$.

Case 1: When candidate trajectories for all rechargeable robots are valid, i.e. $\forall i \in \mathcal{R}$ and $\forall t_{m_1}^{i1}, t_{m_2}^{i2} \in \mathcal{T}$

$$\begin{aligned} x_{j+1}^{i,com}(t) &\leftarrow x_{j+1}^{i,can}(t) \quad \forall t \in [t_{j+1}, t_{j+1,C}^i) \\ &\Rightarrow \begin{cases} T_{F,j+1}^k > (T_R + T_E + kT_\delta) & \forall k \in \mathcal{R}' \\ e^i(t) > e_{j+1}^{res} & \forall t \in [t_{j+1}, t_{j+1,C}^i) \end{cases} \\ &\Rightarrow \begin{cases} |t_{m_1}^{i1} - t_{m_2}^{i2}| > T_\delta & \forall t \in [t_{j+1}, t_{j+1,R}) \\ e^i(t) > e_{min}^i & \forall t \in [t_{j+1}, t_{j+1,R}) \end{cases} \end{aligned}$$

Since $t_{j+1,R} > t_{j,R} \forall i \in \mathcal{R}$ the claim holds.

Case 2: This case corresponds to the scenario when the 0^{th} robot in \mathcal{R}' returns either due to violation of Gap flag condition or the Reserve SoC condition, i.e.,

$$x_{j+1}^{0',com}(t) \leftarrow x_j^{0',com}(t) \quad \forall t \in [t_{j+1}, t_{j,C}^{0'}).$$

The candidate trajectories are committed for the remaining robots, i.e. $\forall k \in \mathcal{R}' \setminus \{0'\}$ and $\forall t_{m_1}^{i1}, t_{m_2}^{i2} \in \mathcal{T}$

$$\begin{aligned} x_{j+1}^{k,com} &\leftarrow x_{j+1}^{k,can} \quad \forall t \in [t_{j+1}, t_{j+1,C}^k) \\ &\Rightarrow \begin{cases} T_{F,j+1}^k > (T_R + T_E + kT_\delta) \\ e^k(t) > e_{j+1}^{res} & \forall t \in [t_{j+1}, t_{j+1,C}^k) \end{cases} \\ &\Rightarrow \begin{cases} |t_{m_1}^{i1} - t_{m_2}^{i2}| > T_\delta & \forall t \in [t_{j+1}, t_{j+1,R}) \\ e^k(t) > e_{min}^k & \forall t \in [t_{j+1}, t_{j+1,R}) \end{cases} \end{aligned}$$

Since $t_{j+1,R} > t_{j,C}^k$, the claim holds. \square