# Adaptive Ergodic Search with Energy-Aware Scheduling for Persistent Multi-Robot Missions

Kaleb Ben Naveed[1*], Devansh R. Agrawal[1], Rahul Kumar[1], Dimitra Panagou[1,2]

[1*]Department of Robotics, University of Michigan, Ann Arbor, 48109, MI, USA.
[2]Department of Aerospace Engineering, University of Michigan, Ann Arbor, 48109, MI, USA.

*Corresponding author(s). E-mail(s): kbnaveed@umich.edu;
Contributing authors: devansh@umich.edu; rahulhk@umich.edu; dpanagou@umich.edu;

**Abstract**

Autonomous robots are increasingly deployed for long-term information-gathering tasks, which pose two key challenges: planning informative trajectories in environments that evolve across space and time, and ensuring persistent operation under energy constraints. This paper presents a unified framework, `mEclares`, that addresses both challenges through adaptive ergodic search and energy-aware scheduling in multi-robot systems. Our contributions are two-fold: (1) we model real-world variability using stochastic spatiotemporal environments, where the underlying information evolves unpredictably due to process uncertainty. To guide exploration, we construct a target information spatial distribution (TISD) based on clarity, a metric that captures the decay of information in the absence of observations and highlights regions of high uncertainty; and (2) we introduce `Robust-meSch` (`RmeSch`), an online scheduling method that enables persistent operation by coordinating rechargeable robots sharing a single mobile charging station. Unlike prior work, our approach avoids reliance on preplanned schedules, static or dedicated charging stations, and simplified robot dynamics. Instead, the scheduler supports general nonlinear models, accounts for uncertainty in the estimated position of the charging station, and handles central node failures. The proposed framework is validated through real-world hardware experiments, and feasibility guarantees are provided under specific assumptions.
[Code: https://github.com/kalebbennaveed/mEclares-main.git]
[Experiment Video: https://www.youtube.com/watch?v=dmaZDvxJgF8]

**Keywords:** Informative path planning, Energy-aware planning, Ergodic search, Multi-agent coordination

## 1 Introduction

Autonomous robots are increasingly deployed in missions requiring long-term data acquisition, such as environmental monitoring (Manjanna, Li, Smith, Rekleitis, and Dudek (2018); Sujit, Sousa, and Pereira (2009)), ocean current characterization (Gawarkiewicz et al. (2018); Todd (2020)), wildfire surveillance (Julian and Kochenderfer (2019)), and search-and-rescue operations (Mayer, Lischke, and Woźniak (2019); Waharte and Trigoni (2010)). Planning informative trajectories for such missions poses two key challenges: (i) designing robot trajectories that maximize information acquisition in dynamic environments, and (ii) ensuring task persistence under energy constraints by enabling timely recharging.

## 1.1 Adaptive Informative Path Planning

The first challenge involves adaptive planning in spatiotemporal environments—where quantities of interest (e.g., temperature, wind speed, gas concentration) evolve across space and time. Informative path planning (IPP) addresses this by generating robot paths that maximize information gain or minimize uncertainty, subject to resource constraints. Classical approaches include orienteering-based formulations (Bottarelli, Bicego, Blum, and Farinelli (2019)), submodular optimization methods (Meliou, Krause, Guestrin, and Hellerstein (2007)), and Gaussian Process (GP)-based planners (Chen, Khardon, and Liu (2022)). To improve adaptability and scalability in high-dimensional settings, sampling-based methods (Moon et al. (2025)) and receding horizon strategies (Sun et al. (2017)) have been proposed. However, many struggle to adapt in real-time to variations in the environment.

Ergodic search offers an alternative by generating trajectories that match the time-averaged visitation frequency with a target information spatial distribution (TISD), instead of choosing discrete sensing points. Prior work (Abraham, Prabhakar, and Murphey (2021); Coffin, Abraham, Sartoretti, Dillstrom, and Choset (2022); Dong, Berger, and Abraham (2023); Dressel and Kochenderfer (2019); G. Mathew and Mezić (2011)) has demonstrated its value in achieving spatially balanced exploration. However, these methods often assume *spatiostatic environments* (Dong et al. (2023); G. Mathew and Mezić (2011)) or rely on known spatiotemporal dynamics (Dressel and Kochenderfer (2019); Garza (2021); Rao et al. (2023)), limiting applicability in real-world scenarios where uncertainty arises from model mismatch, disturbances, or environmental variability.

To address this, we consider *stochastic spatiotemporal environments*—environments whose evolution is uncertain in both space and time. In such cases, information can decay without continued measurement, motivating online trajectory planning that prioritizes regions with high uncertainty and rapid information loss. We build on the clarity metric, proposed by D. R. Agrawal and Panagou (2023), a bounded information measure between $[0, 1]$ that captures both current knowledge and its decay due to lack of observation.

Using clarity, we construct a principled TISD that continuously evolves based on the robot's measurement history and environmental uncertainty, allowing robots to adaptively revisit regions where uncertainty is increasing.

## 1.2 Task Persistence in Multi-Agent Systems

The second challenge is persistent operation under energy constraints, particularly when multiple robots must coordinate recharging through a shared charging resource. Prior work on task persistence spans both single-agent and multi-agent scenarios involving static and mobile charging infrastructure.

For static stations, some methods assume a dedicated charger per robot (Gao and Bhattacharya (2019); Notomista (2022); Notomista, Pacchierotti, and Giordano (2022); Notomista, Ruf, and Egerstedt (2018)), while others support shared chargers with concurrent access (Asghar, Sundaram, and Smith (2023); Kenzin, Bychkov, and Maksimkin (2020)). When fewer chargers than robots are available (Li, Patankar, Moridian, and Mahmoudian (2018); Liu and Michael (2014); Seewald, Lerch, Chancán, Dollar, and Abraham (2024)), strategies include modifying mission paths (Liu and Michael (2014)), placing stations strategically (Li et al. (2018)), or constraining charging frequency (Seewald et al. (2024)). Closest to our work are Bentz, Hoang, Bayasgalan, and Panagou (2018); Fouad and Beltrame (2022): the former staggers robot deployments to ensure exclusivity, while the latter employs control barrier functions (CBFs) Ames, Xu, Grizzle, and Tabuada (2017) to enforce minimum SoC levels under simplified single-integrator dynamics.

Most mobile charging approaches assume a dedicated charging robot, with coordination either via precomputed rendezvous points (Karapetyan et al. (2023); Kingry et al. (2017)) or continuous communication (T. X. Lin, Yel, and Bezzo (2018)). Others dynamically intercept robots during their mission (Couture-Beil and Vaughan (2009); X. Lin, Yazıcıoğlu, and Aksaray (2022); N. Mathew, Smith, and Waslander (2015)). In contrast, we consider a shared mobile charging station that travels alongside the robot network to extend operational time. Our method does not

rely on preplanned rendezvous or continuous communication and supports general nonlinear robot dynamics.

## 1.3 Contributions

This work presents a unified framework for adaptive ergodic search and energy-aware scheduling in persistent multi-robot missions. Our key contributions, situated in the context of existing state-of-the-art methods, are:

- **Principled multi-agent TISD construction via clarity:** Unlike prior ergodic methods that assume static (Dong et al. (2023)) or known spatiotemporal dynamics (Dressel and Kochenderfer (2019); Garza (2021); Rao et al. (2023)), we construct the target information spatial distribution (TISD) using the clarity metric (D. R. Agrawal and Panagou (2023)), a bounded measure that quantifies information decay and the maximum attainable information in stochastic spatiotemporal environments. This allows robots to adaptively focus sensing effort in regions with high uncertainty and rapid information loss.
- **Robust energy-aware scheduling with fail-safe coordination:** Unlike prior work that achieves exclusivity through staggered deployment (Bentz et al. (2018)) or relies on simplified single-integrator dynamics with fixed SoC thresholds (Fouad and Beltrame (2022)), we propose `Robust-meSch` (`RmeSch`), a centralized online scheduling framework that supports general nonlinear robot dynamics, enforces exclusive access to a shared mobile charging station, and guarantees safe returns through a decentralized fail-safe planner that accounts for communication delays and central node failures. Furthermore, we provide formal feasibility guarantees and derive conditions under which robots can be safely added to or removed from the mission without violating energy and return-gap constraints.
- **Hardware-validated multi-agent coordination:** We validate the proposed method on a heterogeneous team comprising multiple aerial robots and a mobile ground-based charging station through extensive hardware experiments.

***Comparison to our own earlier works:*** Compared to our earlier conference papers, this work introduces several key extensions:

- Compared to Naveed, Agrawal, Vermillion, and Panagou (2024a), we extend the clarity-based information model to the multi-agent case, enabling distributed sensing and coordination.
- Compared to Naveed, Dang, Kumar, and Panagou (2024), we introduce a fail-safe planner that enables safe recovery under central node failures and provide a more comprehensive theoretical analysis, including formal guarantees on feasibility and robustness.
- In addition, this paper presents an expanded experimental evaluation compared to both prior works, including real-world demonstrations involving multiple aerial robots coordinating through a shared mobile charging station.

## 2 Preliminaries

### 2.1 Notation

Let $\mathbb{Z}_0 = \{0, 1, 2, ...\}$ and $\mathbb{Z}_+ = \{1, 2, 3, ...\}$. Let $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, $\mathbb{R}_{>0}$ be the set of reals, non-negative reals, and positive reals respectively. Let $\mathbb{S}_{++}^n$ denote set of symmetric positive-definite matrices in $\mathbb{R}^{n \times n}$. Let $\mathcal{N}(\mu, \Sigma)$ denote a normal distribution with mean $\mu$ and covariance $\Sigma \in \mathbb{S}_{++}^n$. The $Q \in \mathbb{S}_{++}^n$, norm of a vector $x \in \mathbb{R}^n$ is denoted $\|x\|_Q = \sqrt{x^T Q x}$. The space of continuous functions $f : \mathcal{A} \to \mathcal{B}$ is denoted as $C(\mathcal{A}, \mathcal{B})$.

### 2.2 System Description

Consider a multi-agent system, in which each robotic system $i \in \mathcal{R} = \{1, \cdots, N\}$, referred to as a ***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 $\mathcal{R}$, $\chi^i = \begin{bmatrix} x^{iT}, e^i \end{bmatrix}^T \in \mathcal{Z}_r^i \subset \mathbb{R}^{n+1}$ 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, $f^i : \mathcal{Z}_r^i \times \mathcal{U}_r^i \to \mathbb{R}^{n+1}$ defines the continuous-time robotic system dynamics, $f_r^i : \mathcal{X}_r^i \times \mathcal{U}_r^i \to \mathbb{R}^n$ 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**):

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

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

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 $W(t) \in \mathbb{R}_{\geq 0}$, $y^c \in \mathbb{R}^c$ 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 $V(t)$.

## 2.3 Ergodic Search

Ergodic search (Dressel and Kochenderfer (2019); G. Mathew and Mezić (2011)) is a technique to generate trajectories $x : [t_0, T] \rightarrow \mathcal{X}$ that cover a rectangular domain $\mathcal{P} = [0, L_1] \times \cdots [0, L_s] \subset \mathbb{R}^s$, matching a specified *target information spatial distribution* (TISD) $\phi : \mathcal{P} \rightarrow \mathbb{R}$, where $s$ is the dimensionality of the environment and $\phi(p)$ is the density at $p \in \mathcal{P}$. Moreover, the spatial distribution of the trajectory $x(t)$ is defined as

$$c(x(t), p) = \frac{1}{T - t_0} \int_{t_0}^{T} \delta(p - \Psi(x(\tau))) d\tau \tag{3}$$

where $\delta : \mathcal{P} \rightarrow \mathbb{R}$ is the Dirac delta function and $\Psi : \mathcal{X} \rightarrow \mathcal{P}$ is a mapping such that $\Psi(x(\tau))$ is the position of the robot at time $\tau \in [t_0, T]$. In other words, given a trajectory $x(t)$, $c(x(t), p)$ represents the fraction of time the robot spends at a point $p \in \mathcal{P}$ over the interval $[t_0, T]$. Then, the *ergodicity* of $x(t)$ w.r.t to a TISD $\phi$ is

$$\Phi(x(t), \phi) = \|c - \phi\|_{H^{-(s+1)/2}} \tag{4}$$

where $\|\cdot\|_{H^{-(s+1)/2}}$ is the Sobolev space norm defined in G. Mathew and Mezić (2011), i.e., $\Phi$ is a function space norm measuring the difference between the TISD $\phi$ and the spatial distribution of the trajectory $c$. Given the ergodic metric, ergodic trajectories for a team of $N$ robots

can be computed by solving the following optimization problem over the space of trajectories $x^i(t) \in C([t_0, T], \mathcal{X})$ and control inputs $u^i(t) \in C([t_0, T], \mathcal{U})$ for each robot $i \in \mathcal{R}$:

$$\min_{\{x^i(t), u^i(t)\}} \quad \Phi(\{x^1(t), \cdots, x^N(t)\}; \phi)$$

$$+ \sum_{i=1}^{N} \int_{t_0}^{T} \|u^i(\tau)\|^2 d\tau$$

$$\text{s.t.} \quad \dot{x}^i = f_r(x^i, u^i), \quad \forall i \in \mathcal{R}$$

$$x^i(t_0) = x_0^i$$

$$\|x^i(t) - x^j(t)\| \geq d_{\min}, \quad \forall i \neq j \tag{5}$$

where $x_0^i$ is the initial state of robot $i$, and $d_{\min}$ is the minimum safety distance to ensure inter-robot collision avoidance. The multi-agent ergodic metric $\Phi(\{x^1(t), \cdots, x^N(t)\}; \phi)$ quantifies the team's collective coverage of the target distribution $\phi$. It is typically computed via a Fourier decomposition of both the empirical visitation statistics and the target distribution (Dressel and Kochenderfer (2019)). This optimization problem can be solved using gradient-based methods. In this work, we do not focus on a specific trajectory optimization method, but rather on the principled construction of the TISD for guiding ergodic exploration in stochastic spatiotemporal environments.

## 2.4 Clarity

We use Clarity D. R. Agrawal and Panagou (2023), an information measure that defines the quality of information about the variable of interest on a $[0, 1]$ scale. Let $X$ be an $n$-dimensional continuous random variable with a density function $\rho(x)$. Its differential entropy is given as follows:

$$h[X] = - \int_S \rho(x) \log \rho(x) dx \tag{6}$$

where $S$ is the support of $X$. Clarity of $X$, derived from differential entropy, is defined as follows:

**Definition 1.** *The Clarity $q[X] \in [0, 1]$ is defined as:*

$$q[X] = \left(1 + \frac{e^{2h[X]}}{(2\pi e)^n}\right)^{-1} \tag{7}$$

4

$q \to 1$ represents the case when $X$ is perfectly known, whereas lower values correspond to higher uncertainty.

Consider a stochastic variable (quantity of interest) $m \in \mathbb{R}$ governed by the process and output (measurement) models:

$$\dot{m} = w(t), \qquad w(t) \sim \mathcal{N}(0, Q) \quad \text{(8a)}$$
$$y = C(x)h + v(t), \qquad v(t) \sim \mathcal{N}(0, R) \quad \text{(8b)}$$

where $Q \in \mathbb{R}_{\geq 0}$ is the known variance associated with the process noise, $y \in \mathbb{R}$ is the measurement, $C : \mathcal{X} \to \mathbb{R}$ is the mapping between robot state and sensor state, and $R \in \mathbb{R}$ is the known variance of the measurement noise.

Clarity $q$ of the random quantity $m$, which lies between $[0, 1]$ and is defined such that $q = 0$ represents $m$ being unknown, and $q = 1$ corresponds to $m$ being completely known. The clarity dynamics for the subsystem (8a), (8b) are given as follows

$$\dot{q} = \frac{C(x)^2}{R}(1 - q)^2 - Qq^2 \quad \text{(9)}$$

# 3 Problem Formulation

In this section, we provide the mathematical formulation of the problem. We first derive the clarity dynamics for multi-robot systems, then describe the environment model, and finally present the overall problem statement.

## 3.1 Multi-robot Clarity Dynamics

We consider the estimation of a scalar stochastic variable $m$ using $N$ robots. The system dynamics are:

$$\dot{m} = w(t), \qquad w(t) \sim \mathcal{N}(0, Q) \quad \text{(10)}$$

Let $X = [x^1, x^2, \ldots, x^N]^T \in \mathbb{R}^{N \times 1}$ denote the stacked state vector of all robots. Each robot $i \in \mathcal{R}$ measures $m$ as follows:

$$y^i = C(x^i)m + v^i(t), \quad v^i(t) \sim \mathcal{N}(0, R(x_i)) \quad \text{(11)}$$

Assuming the measurement noise is independent across agents, the measurements can be stacked as:

$$y(X) = C(X)m + v(X), \quad v(t) \sim \mathcal{N}(0, R(X)) \quad \text{(12)}$$

where

$$C(X) = [C(x^1), C(x^2), \cdots, C(x^N)]^T \in \mathbb{R}^{N \times 1} \quad \text{(13)}$$

$$R(X) = \begin{bmatrix} R(x^1) & 0 & \cdots & 0 \\ 0 & R(x^2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & R(x^N) \end{bmatrix} \in \mathbb{R}^{N \times N} \quad \text{(14)}$$

The Kalman filter equations for the scalar estimate $\mu$ and variance $P$ are:

$$\dot{\mu} = PC(X)^T R(X)^{-1}(y(X) - C(X)\mu) \quad \text{(15a)}$$
$$\dot{P} = Q - PC(X)^T R(X)^{-1} C(X)P \quad \text{(15b)}$$

Since clarity is defined as $q = \frac{1}{1+P}$, the clarity dynamics can be derived as follows:

$$\dot{q} = \frac{-\dot{P}}{(1+P)^2}$$
$$= \frac{1}{(1+P)^2}\left(P^2 C(X)^T R(X)^{-1} C(X) - Q\right) \quad \text{(16)}$$

Substituting $P = \frac{1-q}{q}$, we get

$$\dot{q} = (1-q)^2 C(X)^T R(X)^{-1} C(X) - Qq^2$$
$$= (1-q)^2 \sum_{i \in \mathcal{R}} \frac{C(x^i)^2}{R(x^i)} - Qq^2 \quad \text{(17)}$$

The (17) define the clarity dynamics for the case when measurements from multiple robots are involved in estimating the quantity of interest.

If $C(x^i)$ and $R(x^i)$ are constant for all $i \in \mathcal{R}$, then the clarity dynamics (17) admit a closed-form solution for the initial condition $q(0) = q_0$:

$$q(t; q_0) = q_\infty \left(1 + \frac{2\gamma_1}{\gamma_2 + \gamma_3 e^{2kQt}}\right) \quad \text{(18)}$$

where $k = \sqrt{\frac{\sum_{i \in \mathcal{R}} \frac{C(x^i)^2}{R(x^i)}}{Q}}$, $q_\infty = \frac{k}{k+1}$, $\gamma_1 = q_\infty - q_0$, $\gamma_2 = \gamma_1(k-1)$, and $\gamma_3 = (k-1)q_0 - k$.

As $t \to \infty$, $q(t; q_0) \to q_\infty \leq 1$ monotonically. Thus $q_\infty$ defines the maximum attainable clarity.

Equation (18) can be inverted to determine the time required to increase clarity from $q_0$ to some $q_1$. This time is denoted $\Delta T : [0,1]^2 \to \mathbb{R}_{\geq 0}$:

$$\Delta T(q_0, q_1) = t \text{ s.t. } q(t, q_0) = q_1 \quad \text{for } q_1 \in [q_0, q_\infty) \tag{19}$$

For $q_1 < q_0$, we set $\Delta T(q_0, q_1) = 0$ while $\Delta T(q_0, q_1)$ is undefined for $q_1 \geq q_\infty$.

## 3.2 Environment Specification

Consider the coverage space $\mathcal{P}$. We discretize the domain into a set of $N_p$ cells each with size $V$.[a] Let $m_p : [t_0, \infty) \to \mathbb{R}$ be the (time-varying) quantity of interest at each cell $p \in \mathcal{P}_{\text{cells}} = \{1, ..., N_p\}$. We model the quantities of interest as independent stochastic processes:

$$\dot{m}_p = w_p(t), \qquad w_p(t) \sim \mathcal{N}(0, Q_p) \tag{20a}$$

$$y_p = C_p(X) m_p + v_p(t), \quad v_p(t) \sim \mathcal{N}(0, R(X)) \tag{20b}$$

where $y_p \in \mathbb{R}$ is the output corresponding to cell $p$. $R(X)$ is the measurement noise, and $Q_p \in \mathbb{R}_{>0}$ is the process noise variance at each cell $p$. Since $m_p$ varies spatially and temporally under process noise $Q_p$ for each cell $p \in \mathcal{P}_{\text{cells}}$, the environment becomes a *stochastic spatiotemporal environment*.

## 3.3 Problem Statement

Consider a team of $N + 1$ robots performing persistent coverage of a stochastic spatiotemporal environment (20) over a time horizon $[0, \infty)$. Among them, $N$ robots are rechargeable and require periodic recharging, while one robot serves as a mobile charging robot and does not require recharging.[b] The rechargeable robots model the mobile charging robot using (2). The objectives for the rechargeable robots are twofold:

- Generate nominal informative trajectories for rechargeable robots using clarity-driven ergodic search;

- Ensure mutually exclusive use of the mobile charging robot, which follows a nominal trajectory.

We formulate an optimization problem that captures these objectives. The objective function is designed to maximize clarity across the regions of interest, while constraints ensure that each robot's energy level remains non-negative and that the robots exclusively share the single mobile charging station. We now define the clarity-based objective functional, along with the energy constraints and mutual exclusion constraints related to charging.

### 3.3.1 Clarity-based Objective functional

Assume the desired quality of information at each cell is encoded using a *target clarity* $\overline{q}_p < q_{\infty,p}$ for each cell $p \in \mathcal{P}_{\text{cells}}$. The target clarity can be different at each cell, indicating a different desired quality of information at each cell, but must be less than $q_{\infty,p}$, the maximum attainable clarity of the cell. If $\overline{q}_p \geq q_{\infty,p}$ for a cell $p \in \mathcal{P}_{\text{cells}}$, then the robot would try to spend an infinite amount of time at a cell $p$, which is undesirable.
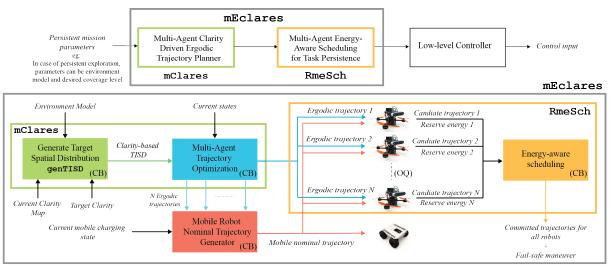
We use clarity as our information metric since it is particularly effective for stochastic spatiotemporal environments:

- The clarity decay rate in cell $p$, i.e. $-Q_p q_p^2$, is explicitly dependent on the stochasticity of the environment $Q_p$ in (20). This allows the information decay rate to be determined from the environment model, and not set heuristically. Furthermore, spatiostatic environments are a special case: by setting $Q_p = 0$, clarity cannot decay.
- While taking measurements of cell $p$, clarity $q_p$ monotonically approaches $q_{\infty,p} < 1$ for $Q_p, R(x^i) > 0, \forall i \in \mathcal{R}$. This indicates that the maximum attainable information is upper bounded.

In this persistent task, the trajectory for each robot is replanned every $T_H \in \mathbb{R}_{>0}$ seconds, i.e., at times $\{t_0, t_1, \cdots\}$ for $t_k = kT_H$, $k \in \mathbb{N}$. At the $k$-th iteration, the objective is to minimize the *mean*

---

**Fig. 1**: `meSch`: The block diagram shows the complete proposed framework `mEclares`.



System Communication Architecture

**RR** Rechargeable robot **MC** Mobile charging robot
**BS** Base Station ◀----▶ Bi-directional communication

**Fig. 2**: The supported communication architecture of the system.

clarity deficit $q_d(t_k + T_H)$, which is defined as

$$q_d(t_k + T_H) = \frac{1}{N_p} \sum_{p=1}^{N_p} \max(0, \overline{q}_p - q_p(t_k + T_H))$$

$$(21)$$

where $q_p(t_k + T_H)$ is the clarity at time $t_k + T_H$ of cell $p \in \mathcal{P}_{\text{cells}}$. However, in order to persistently monitor a stochastic spatiotemporal environment over a long time horizon, the robot's energy constraints must be taken into consideration.

### 3.3.2 Minimum Energy and Mutually Exclusive Charging Constraints

We define $\mathcal{T}^i$ as the set of times $i^{th}$ robot returns to the charging station:

$$\mathcal{T}^i = \{t_0^i, t_1^i, \cdots, t_m^i, \cdots\}, \forall i \in \mathcal{R}, \forall m \in \mathbb{Z}_0 \quad (22)$$

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:

$$e^i(t) \geq e_{min}^i \qquad \forall t \in [t_0, \infty), \forall i \in \mathcal{R} \qquad (23a)$$

$$|t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta \quad \forall t_{m_1}^{i_1}, t_{m_2}^{i_2} \in \mathcal{T} \qquad (23b)$$

Condition (23a), the **minimum SoC condition**, defines the required minimum battery SoC for all rechargeable robots. Condition (23b), 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.

Now we define the optimization problem, which must be solved at times $\{t_0, t_1, \cdots\}$ for $t_k = kT_H$:

**Problem 1.** At each planning time $t_k$, the problem is posed as:

$$\min_{\chi^i(t), u^i(t)} \quad q_d(t_k + T_H) \tag{24a}$$

$$\text{s.t.} \quad \chi^i(t_k) = \chi^i_k, \quad \forall i \in \mathcal{R} \tag{24b}$$

$$\dot{\chi}^i = f(\chi^i, u^i), \quad \forall i \in \mathcal{R} \tag{24c}$$

$$\dot{q}_p = g(x, q_p), \quad \forall p \in \mathcal{P}_{\text{cells}} \tag{24d}$$

$$\left\| x^i(t) - x^j(t) \right\| \geq d_{\min}, \forall i \neq j \tag{24e}$$

$$e^i(t) \geq e^i_{\min}, \quad \forall i \in \mathcal{R} \tag{24f}$$

$$|t^{i_1}_{m_1} - t^{i_2}_{m_2}| > T_\delta, \quad \forall t^{i_1}_{m_1}, t^{i_2}_{m_2} \in \mathcal{T}_{k,H} \tag{24g}$$

where $q_d(t_k + T_H)$ is the mean clarity deficit at the end of system trajectory $\chi^i(t; t_k, \chi_k)$, $\forall t \in [t_k, t_k + T_H]$, $\forall i \in \mathcal{R}$ given by (21), $g : \mathcal{X} \times [0, 1] \to \mathbb{R}_{\geq 0}$ define the clarity dynamics (17), and $e_{min}$ is the minimum energy level allowed for the robot. (24e) defines the collision avoidance constraint for all robots $i, j \in \mathcal{R}$. The set $\mathcal{T}_{k,H} = \mathcal{T} \cap [t_k, t_k + T_H]$ denotes all charging return times within the current planning horizon, and is used to enforce the minimum gap condition over this finite window.

# 4 Method Motivation & Overview

## 4.1 Method Motivation

To solve problem (24), we draw inspiration from ergodic search. As discussed in section 2.3, ergodic search generates trajectories by solving problem (5). When the target information spatial distribution (TISD) $\phi$ is constructed based on the current clarity $q_p(t)$ and a desired target clarity $\bar{q}_p$ at each cell, ergodic search naturally minimizes the mean clarity deficit (21). In this work, we propose a principled method to construct $\phi$ using clarity.

However, the optimization in (5) does not account for energy constraints (23a) or the minimum gap requirement (23b). While one could include these constraints in (5), the non-convexity of the problem makes it difficult to ensure convergence or feasibility. We therefore propose `mEclares`, shown in fig. 1, as an approximate solution to (24).[c]

[c] Note that $q_d(T)$ is not differentiable, making direct optimization of (24) challenging. In contrast, (5) is differentiable

## 4.2 Method Overview

Our approach decouples (24) into two sub-problems: (A) each robot computes an *ergodic trajectory* that maximizes information collection while ignoring energy constraints; (B) each robot then generates a *candidate trajectory* that attempts to track a portion of the ergodic trajectory while reaching the charging station before depleting its energy. All candidate trajectories are sent to the base computer, where the `RmeSch` algorithm evaluates them and decides whether to *commit* each one. Committed trajectories are guaranteed to satisfy the minimum SoC constraint (23a) and the minimum gap constraint (23b). Each robot always tracks its most recent committed trajectory, ensuring persistent exploration while respecting energy constraints and coordinating exclusive access to the mobile charging station. The nominal trajectory of the mobile charging robot is generated so that it travels along the network of rechargeable robots.

These components operate on different timescales. The ergodic trajectory is replanned every $T_H$ seconds, while the committed trajectory is updated every $T_E < T_H$ seconds.[d]

- At each time $t_k = kT_H$, $k \in \mathbb{N}$:
  - Recompute the TISD $\phi$ using `genTISD`.
  - Recompute the ergodic trajectory for the rechargeable robots and the nominal trajectory of the mobile charging robot.

- At each time $t_j = jT_E$, $j \in \mathbb{N}$:
  - Each robot generates a candidate trajectory and sends it to the base computer.
  - The central `RmeSch` algorithm evaluates the candidate trajectories and decides whether to commit each one of them.
  - `RmeSch` also publishes the fail-safe schedule in case the central node fails before the next decision iteration $j + 1$.

Although the proposed method uses centralized decision-making, we distribute computation across the network to enable real-time operation.

and can be efficiently approximated using gradient-based trajectory optimization solvers.

[d] $T_E, T_H \in \mathbb{R}_+$ are user-defined parameters.

**Algorithm 1** The `genTISD` algorithm

1: **function** `genTISD` ($q_p$, $\overline{q}_p$, environment model (20))
2:      **for** $p \in \{1, ..., N_p\}$ **do**
3:          $k \leftarrow \sqrt{\dfrac{\sum_{i \in \mathcal{R}} \frac{C(x^i)^2}{R(x^i)}}{Q}}$
4:          $q_\infty \leftarrow k/(k+1)$
5:          $\overline{q} \leftarrow \min(\overline{q}_p, q_{\infty,p} - \epsilon)$
6:          $\phi_p \leftarrow \Delta T(\overline{q}, q_p)$ using (19)
7:      **end for**
8:      $\phi_p \leftarrow \phi_p / (\sum_{p=1}^{N_p} \phi_p), \quad \forall p \in \{1, ..., N_p\}$
9:      **return** $\phi_p \ \forall p \in \{1, ..., N_p\}$
10: **end function**

fig. 1 provides a high-level view of the architecture, and fig. 2 illustrates two supported communication models. Construction of the TISD, multi-agent ergodic trajectory generation, and the scheduling component of `RmeSch` are executed on a central base computer.

Each rechargeable robot generates a single *candidate trajectory* onboard, which attempts to track a portion of the ergodic trajectory before reaching the charging station. All candidate trajectories are sent to the base computer, where the `RmeSch` algorithm jointly evaluates them and determines which trajectories to commit, based on energy feasibility and coordination requirements. This setup enables decentralized trajectory generation at the robot level while maintaining global coordination through centralized scheduling.

## 4.3 Method Organization

In the next sections, we describe the `mEclares` framework in detail. We begin with `genTISD`, a method for generating the target information spatial distribution (TISD) used in multi-agent ergodic search. We then present the details of the `RmeSch` algorithm. We also establish notation for trajectories. Let $x^i([t_k, t_k + T_H]; t_k, x_k^i)$ represent the ergodic trajectory for the $i^{\text{th}}$ rechargeable robot at time $t_k$, starting from state $x_k^i$ and defined over a time horizon of $T_H$ seconds. We denote this as $x_k^{i,\text{ergo}}$. The same notation applies to other trajectories. An overview of the notation is provided in table 1. Without loss of generality, we present our method assuming $N$ rechargeable robots modeled as quadrotors and one mobile charging rover.

# 5 Generate Target Spatial Distribution (`genTISD`)

The `genTISD` algorithm is described in algorithm 1. Let $\phi_p$ denote the target information density evaluated for cell $p$. At the $k$-th iteration (i.e, at time $t_k = kT_H$), we set $\phi_p$ to be the time that the robot would need to increase the clarity from $q_p(t_k)$ to the target $\overline{q}_p$ by observing cell $p$ (Lines 3-6). This is determined using (19). The small positive constant $\epsilon > 0$ in Line 5 ensures that target clarity is always less than the maximum attainable clarity, i.e., $\overline{q}_p < q_{\infty,p}$. Finally, we normalize $\phi_p$ such that the sum of $\sum_{p \in \mathcal{P}_{\text{cells}}} \phi_p = 1$ (Line 8). Once $\phi$ is constructed, trajectory optimization solvers can be used to generate the ergodic trajectories $x_k^{i,ergo}, \forall i \in \mathcal{R}$.

# 6 Mobile Charging Station Nominal Trajectory

To support coordination with the team of rechargeable robots, we generate a nominal trajectory for the mobile charging station that tracks the geometric center of the team's nominal ergodic trajectories. At each decision point $t_k$, the geometric center of the team's ergodic trajectories is defined as:

$$x^{\text{cent}}(t) = \frac{1}{N} \sum_{i=1}^{N} x_k^{i,\text{ergo}}(t), \quad \forall t \in [t_k, t_k + T_H]. \tag{25}$$

At time $t_k$, the mobile charging nominal trajectory $x_k^{c,nom}$, defined over the time interval $[t_k, t_{k,H}]$, is generated by solving the following optimal control problem:

$$\min_{x^c(t), u^c(t)} \int_{t_k}^{t_{k,H}} \left\| x^c(t) - x^{\text{cent}}(t) \right\|_{\mathbf{Q}}^2 + \| u^c(t) \|_{\mathbf{R}}^2 \, dt \tag{26a}$$

$$\text{s.t.} \quad x^c(t_k) = x_k^{c,nom}(t_k) \tag{26b}$$

$$\dot{x}^c = f_r^c(x^c, u^c) \tag{26c}$$

where $\mathbf{Q} \in \mathbb{S}_{++}^n$ and $\mathbf{R} \in \mathbb{S}_{++}^m$ weights state cost and control cost respectively. This formulation ensures that the mobile charging robot stays centrally positioned relative to the rechargeable robots without requiring explicit communication

or coordination, enabling robust support for persistent operation.

# 7 Robust Multi-Agent Energy-Aware Scheduling for Task Persistence (RmeSch)

To facilitate readers, we organize the presentation of RmeSch into three subsections: section 7.1 introduces the motivation behind RmeSch and outlines its key ideas, section 7.2 describes the method in detail, and section 7.3 discusses theoretical guarantees around RmeSch.

## 7.1 RmeSch Motivation and Key Ideas

As a low-level module, RmeSch ensures task persistence. The solution follows three steps, with the RmeSch 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 required at the rendezvous to account for uncertainty in the charging station's position.
- Construct a trajectory that follows a portion of the ergodic trajectory before reaching the rendezvous point. We refer to this as the *candidate trajectory*.
- Commit the candidate trajectory if it satisfies both the minimum energy condition (23a) and the minimum gap condition (23b). The result is the *committed trajectory*.
- Along with the committed trajectory, a fail-safe return schedule is generated based on the current SoC level to ensure safe return in case of central node failure.

Before detailing each step, we first explain how RmeSch evaluates the satisfaction of conditions (23a) and (23b). This is one of our key contributions, and we explain it by first discussing its motivation and then describing its mechanism.

### 7.1.1 RmeSch Motivation

Consider $N$ quadrotors sharing a mobile charging rover, as shown in fig. 3. To prevent charging conflicts, we propose a scheduling method based on two principles.
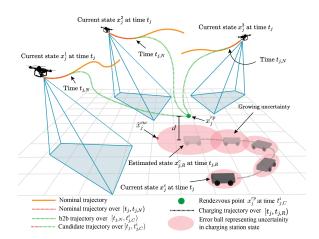


**Fig. 3**: This figure illustrates the generation of candidate trajectories at time $t_j$. All the candidate trajectories terminate at the rendezvous point $x_j^{rp}$ at time $t_{j,C}^i$.

First, if multiple robots are predicted to arrive simultaneously, one is rescheduled to arrive earlier using gap flags explained below. Second, if robots 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 enforces the minimum time gap between consecutive charging sessions by constructing ***gap flags*** and resolving conflicts by selecting the robot with the least remaining flight time to return first—similar to dropping a constraint to restore feasibility. This allows the remaining robots to maintain the desired gap defined by (23b). Once the gap flags are satisfied, the eware module checks whether each robot has enough energy to continue its mission, ensuring that the minimum energy condition (23a) is also satisfied.

### 7.1.2 Key Idea: Construction of Gap flags

We begin by describing the construction of *gap flags* and their role in preventing charging conflicts. At each iteration of RmeSch, rechargeable robots are sorted by their remaining flight time into the ordered set $\mathcal{R}' = \{1', \ldots, N'\}$, where $1'$ has the least flight time. For each robot $l \in$

| Symbol | Definition |
|---|---|
| **Indices** | |
| $i$ | Rechargeable robot index |
| $j$ | `RmeSch` iteration index |
| $k$ | Nominal trajectory planner iteration index |
| $l$ | Rechargeable robot index in sorted list |
| **Constant shared time horizons** | |
| $T_\delta$ | $T_\delta = T_{ch} + T_{bf}$ 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 ($T_{B,j}^i = T_{C,j}^i - T_N$) |
| $T_{F,j}^l$ | Remaining battery time of the $l^{th}$ 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 1**: Time and Index Notation at a glance

$\mathcal{R}' \setminus \{1'\}$, a gap flag is constructed relative to $1'$ as:

$$G^l = T_{F,j}^l > (T_R + T_E + lT_\delta), \qquad (27)$$

where $T_{F,j}^l$ is $l^{th}$ 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 $T_\delta$ includes the charging duration and the buffer time required to resume the mission.

These flags enforce a minimum gap of $lT_\delta$ between robot $1'$ and robot $l$ 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., $1'$, is rescheduled for recharging. The satisfaction of the gap flag condition guarantees that there will be at least $T_\delta$ between successive charging sessions.

## 7.2 `RmeSch` Methodology

In this section, we present `RmeSch` in detail. After establishing the construction of gap flags, we demonstrate how they are iteratively checked within the full solution scheme to ensure conditions (23a) and (23b) 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 `RmeSch`, the nominal trajectories of the rechargeable robots are known for $T_N$ seconds, and the nominal trajectory of the mobile charging robot's for $T_R$ seconds. This can be ensured by selecting time horizons such as they satisfy $T_N, T_R < T_H$.

### 7.2.1 Estimating Rendezvous Point

At the $j^{th}$ iteration of `RmeSch`, 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 fig. 3.

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 Gelb et al. (1974) 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:

$$x_j^{rp} = \begin{bmatrix} \Psi(\hat{x}^c(t_{j,R})) \\ \mathbf{0}_{n-2} \end{bmatrix} + \begin{bmatrix} \mathbf{0}_2 \\ d \\ \mathbf{0}_{n-3} \end{bmatrix} \qquad (28)$$

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

### 7.2.2 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:

$$\hat{x}_j^{cw} = \hat{x}^c(t_{j,R}) + q_{max}\sqrt{\chi_{c,0.95}^2 \lambda_{max}} \tag{29}$$

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 $\lambda_{max}$, 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 $\forall i \in \mathcal{R}$:

$$\min_{\chi^i(t), u^i(t), t_f^i} \quad t_f^i \tag{30a}$$

$$\text{s.t.} \quad \chi^i(t_0^i) = \chi_{rp}^i \tag{30b}$$

$$\dot{\chi} = f_r^i(\chi^i, u^i) \tag{30c}$$

$$x^i(t_f^i) = \hat{x}_j^{cw} \tag{30d}$$

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:

$$e_j^{i,res} = e^i(t_f^i) - e^i(t_0^i) \quad \forall i \in \mathcal{R} \tag{31a}$$

$$T_{L,j}^i = t_f^i - t_0^i \quad \forall i \in \mathcal{R} \tag{31b}$$

### 7.2.3 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

**Algorithm 2** The `RmeSch` algorithm

1: **function** RmeSch($x_j^{i,can}$, $x_{j-1}^{i,com}$, $e_j^{i,res}$)
2:    **if** $x_j^{i,can}$, $x_{j-1}^{i,com}$, $e_j^{i,res}$ not received for all $i \in \mathcal{R}$ **then**
3:       **return** RmeSch($x_j^{i,can}$, $x_{j-1}^{i,com}$, $e_j^{i,res}$)
4:    **end if**
5:    GapVio, $x_j^{i,com}$, $\mathcal{R}' \leftarrow$ gware($x_j^{i,can}$, $x_{j-1}^{i,com}$)
6:    $ret_j^i \leftarrow$ index($i, \mathcal{R}'$)   s.t. $\mathcal{R}'[l] = i$   ▷ Index of robot $i$ in $\mathcal{R}'$
7:    **if** GapVio $== 1$ **then**
8:       **Publish** mobcon$_j$ = True  ▷ publishes message to mobile charging station to continue the mission
9:       **return** $x_j^{i,com}, l^i$   $\forall i \in \mathcal{R}$
10:   **end if**
11:   $x_j^{i,com} \leftarrow$ eware($x_j^{i,can}$, $x_{j-1}^{i,com}$, $e_j^{i,res}$)
12:   **Publish** mobcon$_j$ = True
13:   **return** $x_j^{i,com}, ret_j^i$   $\forall i \in \mathcal{R}$
14: **end function**

at $t_j$ be $\mathcal{X}_j^i \in \mathcal{Z}_r^i$. We construct a b2b trajectory $x_j^{i,b2b}$ defined over interval $[t_{j,N}, t_{j,C}^i]$ by solving:

$$\min_{x^i(t), u^i(t)} \int_{t_{j,N}}^{t_{j,C}^i} \left\| x^i(t) - x_j^{rp} \right\|_{\mathbf{Q}}^2 + \left\| u^i(t) \right\|_{\mathbf{R}}^2 dt \tag{32a}$$

$$\text{s.t.} \quad x^i(t_{j,N}) = x_j^{i,nom}(t_{j,N}) \tag{32b}$$

$$\dot{x}^i = f_r^i(x^i, u^i) \tag{32c}$$

$$x^i(t_{j,C}^i) = x_j^{rp} \tag{32d}$$

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

$$\chi_j^{i,can} = \begin{cases} x_j^{i,can}(t), & t \in [t_j, t_{j,C}^i) \\ e_j^{i,can}(t), & t \in [t_j, t_{j,C}^i) \end{cases} \tag{33}$$

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

$$\dot{\chi}^i = f(\chi^i, u^i(t)), \tag{34a}$$

$$\chi^i(t_j) = \chi_j^i \tag{34b}$$

$$u^i(t) = \begin{cases} \pi_r^i(\chi^i, x_j^{i,nom}(t)), & t \in [t_j, t_{j,N}) \\ \pi_r^i(\chi, x_j^{i,b2b}(t)), & t \in [t_{j,N}, t_{j,C}^i) \end{cases} \tag{34c}$$

where $\pi_r^i : \mathcal{Z}_r^i \times \mathcal{X}_r^i \to \mathcal{U}_r^i$ is a control policy to track the portion of the nominal trajectory and the b2b trajectory. Figure 3 shows the candidate

**Algorithm 3** The `gware` algorithm

1: **function** gware($x_j^{i,\text{can}}$, $x_{j-1}^{i,\text{com}}$)
2:     Sort $x_j^{i,\text{can}}$ based on $T_F^i$
3:     **for** $l \in \mathcal{R}' \setminus \{1'\} = \{2', \ldots, N'-1\}$ **do**
4:         $G^l \leftarrow (T_F^l - T_R - T_E) > l(T_\delta)$
5:         **if** $G^l == 0$ **and** $l.\text{charging} \neq 1$ **then**
6:             $x_j^{1',\text{com}} \leftarrow x_{j-1}^{1',\text{com}}$
7:             $x_j^{l,\text{com}} \leftarrow x_j^{l,\text{can}}$ for all $l \in \mathcal{R}'$
8:             **return** True, $x_j^{i,\text{com}}$, $\mathcal{R}'$
9:         **end if**
10:     **end for**
11:     **return** False, $x_j^{i,\text{com}}$, $\mathcal{R}'$
12: **end function**

---

**Algorithm 4** The `eware` algorithm

1: **function** eware($x_j^{i,\text{can}}$, $x_{j-1}^{i,\text{com}}$, $e_j^{i,\text{res}}$)
2:     **for** $i \in \{1, \ldots, N\}$ **do**
3:         **if** $e^i(t) \geq e_j^{i,\text{res}} \quad \forall t \in [t_j, t_{j,C}^i]$ **then**
4:             $x_j^{i,\text{com}} \leftarrow x_j^{i,\text{can}}$
5:         **else**
6:             $x_j^{i,\text{com}} \leftarrow x_{j-1}^{i,\text{com}}$
7:         **end if**
8:     **end for**
9: **end function**

---

trajectory generation process with 3 rechargeable robots and 1 mobile charging robot.

### 7.2.4 Robust 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 (23a) and the minimum gap condition (23b) are satisfied throughout the candidate trajectory. The overall algorithm described in algorithm 2 consists of the two subroutines: `gware` (gap-aware) and `eware` (energy-aware).

### 7.2.5 Gap-aware (`gware`)

`gware` described in algorithm 3 checks if the rechargeable robots would continue to have the gap of $T_\delta$ seconds between their expected returns if the candidate trajectories were committed.

(Lines 2-4) Here the gap flags are constructed for each $l^{th}$ robot that is not currently charging or returning, relative to the first robot in the sorted list (the $1^{st}$ robot)

$$G^l = T_{F,j}^l > (T_R + T_E + lT_\delta) \tag{35}$$

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

$$T_{F,j}^l > (T_R + T_E + lT_\delta) \tag{36}$$
$$\implies |t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta \qquad \forall t \in [t_j, t_{j,R}) \tag{37}$$

(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.

### 7.2.6 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 4.

(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**:

$$e^i(t) > e_j^{i,res} \ \forall t \in [t_j, t_{j,C}^i] \tag{38}$$

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 $\pi_l^i : [t_{j,C}^i, t_{j,R}) \times \mathcal{X}_r^i \to \mathcal{U}$ that guides the rechargeable robot to the mobile charging robot.

### 7.2.7 Fail-safe maneuver planning

At each iteration of `RmeSch`, we also plan a fail-safe maneuver to handle scenarios involving central node failure or communication delays. Specifically, `RmeSch` transmits to each rechargeable robot a message indicating whether to commit the new trajectory. Along with this, each robot is assigned

**Algorithm 5** Fail-safe onboard rechargeable robot $i$

1: **Try** $(x_j^{i,\text{com}},\ \text{ret}_j^i) \leftarrow \text{RmeSch}(x_j^{i,\text{can}},\ x_{j-1}^{i,\text{com}},\ e_j^{i,\text{res}})$ until $t_{j-1,N}$
2: **if** successful **then**
3:     Execute $x_j^{i,\text{com}}$
4: **else**
5:     **if** $\text{ret}_{j-1}^i == 1$ **then**
6:         Execute $x_{j-1}^{i,\text{com}}(t)$ over $[t_j,\ t_j + T_R]$
7:     **else**
8:         Idle (hover) for time $\text{ret}_{j-1}^i(T_\delta)$
9:         Execute $x_{j-1}^{i,\text{com}}(t - \text{ret}_{j-1}^i(T_\delta))$ over time horizon $[t_j + \text{ret}_{j-1}^i(T_\delta),\ t_j + \text{ret}_{j-1}^i(T_\delta) + T_R]$
10:     **end if**
11: **end if**

---

**Algorithm 6** Fail-safe onboard mobile charging robot

1: At time $t_j$ **Initialize** $\text{mobcontinue}_j$ = False
2: **Try** Message $\text{mobcontinue}_j$ received until $t_{j-1,N}$
3: **if** successful **then**
4:     **if** $\text{mobcontinue}_j$ == True **then**
5:         **Continue** executing the nominal trajectory
6:     **end if**
7: **else**
8:     Execute stopping at time $t_j + T_R$
9: **end if**

---

a return position in the sorted list $\mathcal{R}'$, which is generated based on remaining flight time. This position provides each robot with its rank in the return sequence in case no message is received due to failure. The logic executed onboard each rechargeable robot $i$ is detailed in algorithm 5 and the fail-safe logic for the mobile charging robot is detailed in algorithm 6. Figure 4 illustrates the behavior of the fail-safe maneuver.

*1) Fail-safe maneuver onboard rechargeable robot* (Lines 1–3) Once the candidate trajectories are generated onboard, each rechargeable robot transmits its candidate trajectory to the base computer, which executes RmeSch to determine whether the trajectory should be committed. Starting at time $t_j$, each robot awaits a response from the base until $t_{j-1,N}$. Nominally, the robot should receive this message by $t_{j,T+E} < t_{j-1,N}$; however, due to potential communication delays, a response may arrive later. If a valid response is received by $t_{j,N}$, the robot executes the new committed trajectory.

(Lines 4–9) If no message is received from the central node by time $t_{j,N}$, robot $i$ executes a fallback maneuver using its previously committed trajectory $x_{j-1}^{i,\text{com}}$ and its return index $\text{ret}_{j-1}^i$ from the previous decision epoch. If $\text{ret}_{j-1}^i = 1$,

the robot continues executing its last committed trajectory:

$$x_j^{i,\text{com}}(t) = x_{j-1}^{i,\text{com}}(t), \quad t \in [t_j,\ t_j + T_R] \quad (39)$$

If $\text{ret}_{j-1}^i > 1$, the robot remains idle (e.g., hovers) for $\text{ret}_{j-1}^i(T_\delta)$ seconds, and then begins executing a time-shifted version of its previous trajectory:

$$x_j^{i,\text{com}}(t) = x_{j-1}^{i,\text{com}}(t - \text{ret}_{j-1}^i T_\delta),$$
$$t \in [t_j + \text{ret}_{j-1}^i T_\delta,\ t_j + \text{ret}_{j-1}^i T_\delta + T_R] \quad (40)$$

This fallback guarantees mutually exclusive access to the charging station and ensures energy-feasible operation even in the absence of centralized coordination.

*2) Fail-safe maneuver onboard mobile charging robot*

To detect central node failures, the mobile charging robot monitors the $\text{mobcontinue}_j$ message. If this message is received by $t_{j-1,T_N}$, the robot continues executing its nominal trajectory. Otherwise, if the message is not received by the deadline, it halts the mission at time $t_{j,R}$.

## 7.3 RmeSch Theoretical Guarantees

This section provides the theoretical conditions under which RmeSch guarantees feasibility, robustness, and adaptability. We begin by deriving an upper bound on the number of robots that can be supported at mission start based on the minimum remaining flight time. We then present a general feasibility theorem that guarantees all robots return safely with the required energy and time separation under both nominal conditions and central node failure. Finally, we discuss robustness by addressing two key scenarios: rechargeable robot failure and the addition of new robots during the mission, and provide conditions under which feasibility is preserved in both cases.

### 7.3.1 Upper bound on number of robots

**Lemma 1.** *At iteration $j = 0$, given the sorted list of remaining flight times $\{T_{F,0}^{1'}, T_{F,0}^{2'}, \dots\}$ where $T_{F,0}^{1'}$ is the minimum remaining flight time, the maximum number of robots that can be safely supported by the mission while satisfying all gap*
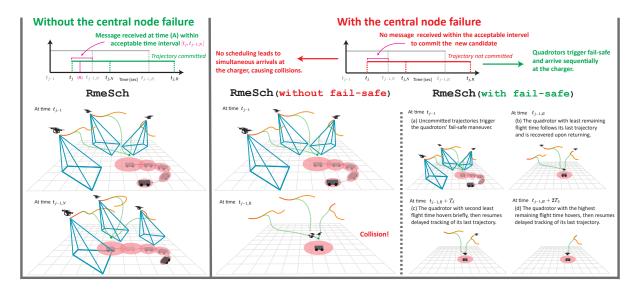
**Fig. 4**: The figure shows the behavior of the algorithm when the central node fails.

*flags is*

$$N^* = 1 + \left\lfloor \frac{T_{F,0}^{1'} - T_R - T_E}{T_\delta} \right\rfloor, \qquad (41)$$

*where $T_R$ is the time a rechargeable robot takes to reach the charging station, $T_E$ is the iteration interval, and $T_\delta$ is the minimum required gap between two consecutive returns.*

*Proof* Please see Appendix A Proofs section A.1 for detailed proof. □

### 7.3.2 Feasibility guarantees

**Theorem 1.** *Given $|\mathcal{R}| \leq N^*$ derived in Lemma 1, suppose that at $j = 0$, the Gap flag condition (35) and the Reserve SoC condition (38) are satisfied. Then, the minimum energy constraint (23a) and the return gap condition (23b) hold for all $t \in [t_0, t_{0,R}^i)$. For all subsequent iterations $j \geq 1$, if solutions for (30) and (32) exist, and the committed trajectories $x_j^{i,com} = x^i([t_j, t_{j,C}^i]; t_j, x_j^i)$ are computed for all $i \in \mathcal{R}$ using algorithm 2, algorithm 5 and algorithm 6, then the conditions (23a) and (23b) are satisfied for all $t \in [t_j, t_{j-1,R})$ and for all $j \in \mathbb{Z}_+$.*

*Proof* Please see Appendix A Proofs section A.2 for detailed proof. □

### 7.3.3 Robustness to rechargeable robot failures and additions

**Remark 1.** Given Lemma 1, Theorem 1 also applies in the case when a subset of robots in $\mathcal{R}$ fail during the mission execution. If such failures are detected and the corresponding robots are excluded from future gap flag evaluations, then the minimum energy condition (23a) and the minimum return gap condition (23b) continue to hold for the remaining robots.

**Remark 2.** Following from lemma 1, at any decision iteration $j$, a new robot can be safely added to the mission if the minimum remaining flight time satisfies

$$T_{F,j}^{1'} \geq T_R + T_E + (N_{\text{curr}}) \cdot T_\delta, \qquad (42)$$

where $T_{F,j}^{1'}$ is the minimum remaining flight time at time $t_j$ and $N_{\text{curr}}$ is the number of robots currently in the mission. This condition ensures that the updated team remains within the admissible bound derived in lemma 1 and all robots satisfy the return gap (23b) and energy (23a) constraints.

## 8 Results & Discussion

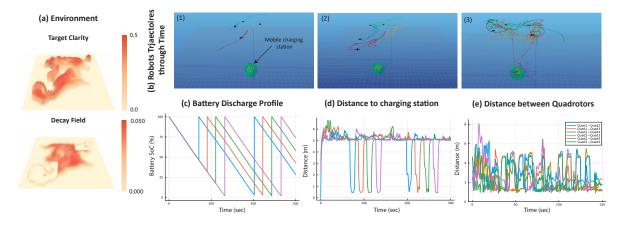In this section, we evaluate `mEclares` through case studies, baseline comparisons, and hardware

15

**Fig. 5**: Demonstration of `mEclares` through a case study: the rechargeable quadrotors track ergodic trajectories that are replanned every 30 seconds, while the mobile charging rover follows the geometric center of the nominal ergodic trajectories of all rechargeable robots.

experiments. We use quadrotors with 3D nonlinear dynamics from (Jackson, Tracy, & Manchester, 2021, 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.

To generate b2b trajectories, we solve (32) using MPC with the reduced linear quadrotor dynamics from Jackson et al. (2021). We use an LQR controller for (30) 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.

## 8.1 Multi-Agent Energy-Aware Persistent Ergodic Search

We evaluate `RmeSch` by simulating a scenario in which four quadrotors and one rover explore a $10 \times 10$ m domain. The nominal trajectories are collision-free, ergodic paths with a horizon of $T_H = 30.0$ s. All quadrotors follow discharge dynamics given by $\dot{e} = -0.667$. Figure 5 (a) shows the target clarity distribution and the decay field of the test environment, where the decay field corresponds to the $Q$ values across the domain, as defined in (17). Figure 5 (b) presents still frames from the lightweight UAV simulator, where four quadrotors explore a stochastic spatiotemporal environment. The mobile charging rover tracks

the geometric center of the four rechargeable quadrotors.

Figure 5 (c) illustrates the battery discharge profiles of the quadrotors, while Figure 5 (d) shows the distance of each quadrotor to the charging station over time. The results indicate that the quadrotors maintain the minimum required gap between successive visits to the charging station. Collision avoidance is implemented using a potential field method, which generates artificial repulsive forces to steer robots away from each other in real time. To ensure that no more than two quadrotors are on charging-related paths at the same time, the $T_\delta$ parameter is set such that a quadrotor returning from the charging station has sufficient time to rejoin the mission before another begins its return to the charger. Finally, Figure 5 (e) shows the inter-quadrotor distances, confirming that no collisions occur during the mission.

## 8.2 Multi-agent Clarity-driven ergodic planner performance comparison to baseline methods

We compare the performance of the proposed method `genTISD` against two baseline strategies. The first baseline is a lawnmower coverage path Choset and Pignon (1998), and the second is the standard uniform TISD used in most ergodic literature Dong et al. (2023); G. Mathew and Mezić (2011).
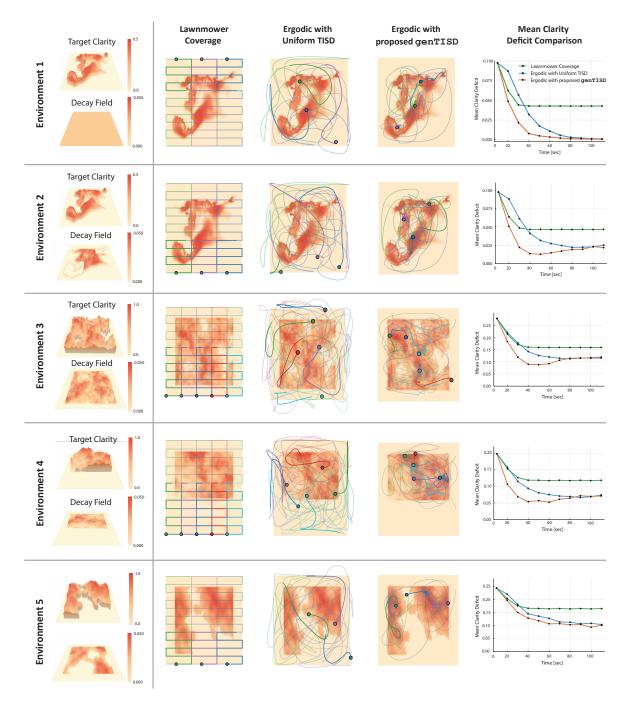
**Fig. 6**: Comparison of mean clarity deficit over time across five synthetic environments.

Performance is evaluated across five different synthetic environments. Each environment specifies a target clarity distribution and a decay field, where the decay field represents the value of $Q$ across the domain, as defined in (17).

The results, summarized in Figure 6, show that the proposed `genTISD` consistently achieves a lower mean clarity deficit compared to both baseline methods across all environments. Specifically, the lawnmower coverage strategy exhibits

**Table 2**: Comparison of baseline methods and proposed `RmeSch`

| Method | Robot Model (Supports Nonlinear Dynamics) | Total Recharging Visits | Gap Violations | Min Energy Violations | Scalability Analysis | Staggered Deployment | Mobile Charging | Central Node failure |
|---|---|---|---|---|---|---|---|---|
| Baseline 1 | SI (No) | 8 | 0 | 0 | Not provided | No | No | No |
| Baseline 2 | Quadrotor (Yes) | 8 | 0 | 0 | Not provided | Yes | No | No |
| Baseline 3 | Quadrotor (Yes) | 8 | 2 | 0 | Not provided | No | No | No |
| Baseline 4 [meSch with only gware] | Quadrotor (Yes) | 4 | 0 | 4 | $\mathcal{O}(N \log N)$ | No | Yes | No |
| Baseline 5 [meSch] | Quadrotor (Yes) | 8 | 0 | 0 | $\mathcal{O}(N \log N)$ | No | Yes | No |
| **Proposed [RmeSch]** | **Quadrotor (Yes)** | 8 | 0 | 0 | $\mathcal{O}(N \log N)$ | No | Yes | Yes |

relatively poor performance, especially in environments with nonuniform decay, as it does not adapt to spatial variations in target clarity or information decay rates. The ergodic control using uniform TISD improves over lawnmower coverage by distributing effort more evenly; however, it still fails to prioritize regions according to their target clarity. In contrast, `genTISD` dynamically allocates exploration effort toward regions with faster clarity decay or higher target clarity, resulting in more efficient information acquisition and lower overall clarity deficits over time.

It is important to note that in Environments 2 through 5, the mean clarity deficit does not converge to zero. This behavior is expected because the decay field $Q$ is non-zero in these environments, causing the target clarity to continuously degrade over time. As a result, it is not possible to achieve perfect target clarity even under optimal exploration. In contrast, Environment 1 has a decay field with $Q = 0$ everywhere, allowing the robots to eventually drive the mean clarity deficit to zero through persistent exploration.

### 8.3 `RmeSch` performance comparison to baseline methods

We compare `RmeSch` to baseline methods using eight metrics, as shown in table 2. For each method, four robots are used with the 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
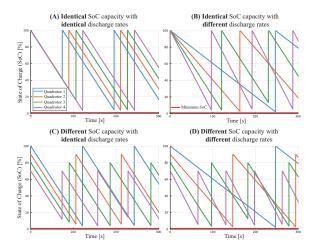


**Fig. 7**: These plots show results for the scenarios when four quadrotors have different SoC capacities and different discharge rates. The plots validate that quadrotors always maintain the minimum of $(T_\delta)$ gap while visiting the charging station.

robot visits and does not account for the minimum energy requirements.

Compared to Baseline 1 (Fouad and Beltrame (2022)), `RmeSch` supports nonlinear dynamic models, making it more applicable to real-world robotic platforms, as demonstrated with 3D quadrotor dynamics Jackson et al. (2021). Unlike Baseline 2 (Bentz et al. (2018)), `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
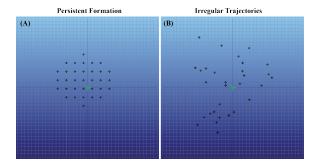
**Fig. 8**: `RmeSch` scalability is demonstrated through two simulation case studies: (A) 30 quadrotors performing a persistent mission, and (B) 30 quadrotors following irregular trajectories.

the number of robots available for the mission at any given moment, limiting overall efficiency. By allowing all robots to be deployed simultaneously, `RmeSch` simplifies mission planning and increases adaptability to different discharge patterns, as shown in fig. 7 with four quadrotors. Compared to Baseline 3 (Naveed, Agrawal, Vermillion, and Panagou (2024b)), `RmeSch` eliminates simultaneous charging station visits. In Baseline 3, four robots returned concurrently on two occasions, leading to a violation of (23b). While Baseline 4, which only includes the `gware` module from `meSch` (`RmeSch` without fail-safe planner), successfully avoids overlapping visits, it fails to enforce minimum energy constraints, resulting in a violation of (23a). Finally, none of the baseline methods support mobile charging stations—a limitation in environments where fixed charging locations may be infeasible. They also lack safe recovery maneuvers in the event of a central node failure. By addressing these gaps, `RmeSch` enhances mission endurance and scalability while providing provable safety and feasibility guarantees.

### 8.3.1 Computational efficiency and scalability

Distributing computation across the robot network improves the efficiency of the `RmeSch` module. The main overhead comes from generating candidate trajectories, with solving (32) and integrating the system's nonlinear dynamics taking 150 ms and 30 ms on average, respectively. We employ the communication architecture(s) shown in fig. 2.

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 of algorithm 3. 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 30 rechargeable quadrotors as shown in fig. 8. In these simulations, quadrotors return with $(3 \pm 1)\%$ battery SoC remaining.

### 8.3.2 Hardware Demonstration

We validate `mEclares` through a set of real-world hardware experiments involving rechargeable quadrotors and a mobile charging rover. Each quadrotor runs onboard computation on an NVIDIA Orin NX, while the rover uses a Raspberry Pi. The communication architecture used in these experiments is shown in fig. 2. All experiments were conducted in the FlyLab facility at Michigan Robotics—a three-floor indoor arena equipped with 15 Vicon cameras for high-precision state estimation.

In all experiments, only the next $T_N = 2.0$ seconds of the nominal trajectory is provided by the high-level planner. Candidate trajectories are generated onboard each quadrotor and transmitted to the base station computer, which verifies gap flags and minimum state-of-charge (SoC) conditions. The rover (when mobile charging is used) continuously publishes its own state and nominal trajectory to support trajectory generation. The experiments highlight three key aspects of our framework:

- the ability to generate ergodic trajectories online in real time,
- the ability to generate candidate trajectories onboard each quadrotor at 1.0 Hz using only 2.0 seconds of the available nominal trajectory and validate them at a central node, and
- the modularity of `RmeSch`, which functions as a low-level scheduling module that remains effective even when the high-level planner is replaced with a non-ergodic coverage strategy.

Experiments are summarized in fig. 9. In the first set of experiments (Experiments 1–2), the
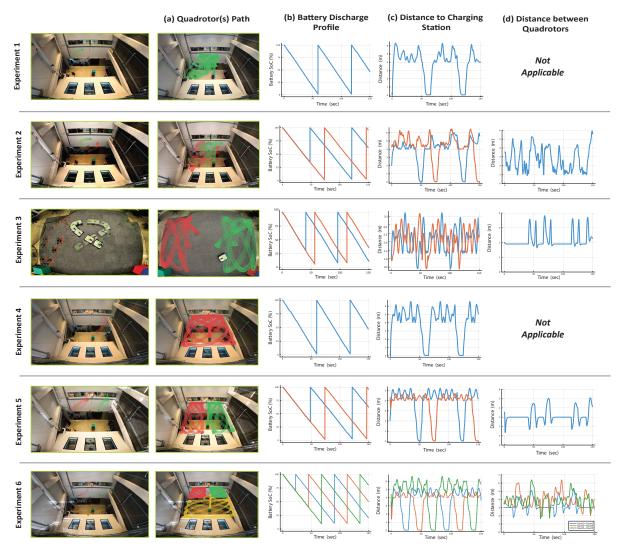
**Fig. 9**: Demonstration of `mEclares` and `RmeSch` on hardware

quadrotors track ergodic trajectories that are replanned every 30 seconds. These trials validate that ergodic exploration and energy-aware scheduling can operate in tandem under real-world conditions. The target clarity for this set of experiments corresponds to Environment 2 in fig. 6, where the quadrotors are observed to spend more time in regions with higher clarity deficit.

In Experiment 3, we demonstrate the use of a mobile charging station. We also show that the charging rover's path can be changed to a

Lissajous curve, and the framework still functions correctly—highlighting the flexibility of the `mEclares` design.

Experiments 4–6 evaluate `RmeSch` under a non-ergodic high-level planner. In these experiments, the quadrotors follow Lissajous coverage trajectories. Candidate trajectories are generated onboard every second and validated at the central node. `RmeSch` continues to ensure safe and effective scheduling under this design.

Collision avoidance is implemented in all experiments using a potential field method, which

20

generates artificial repulsive forces in real time to prevent inter-robot collisions.

Figure 9 (a) shows the coverage paths followed by the quadrotors. fig. 9 (b) and (c) present the battery discharge profiles and distances to the charging station, respectively, confirming that robots never violate the minimum energy requirement (which is zero) and consistently satisfy the minimum desired gap requirement between charging returns. Finally, fig. 9 (d) confirms that no collisions occur during the experiments.

Our implementation also accounts for delays introduced by computational overhead and ROS2 message latency. The primary sources of delay include candidate trajectory generation and forward propagation ($T_1$), gap flag construction and verification ($T_2$), and message latency in ROS2 ($T_3$). As long as $T_1 + T_2 + T_3 < T_E$, where $T_E$ is the `RmeSch` decision interval, the mission proceeds as intended. If these delays exceed the worst-case allowed duration, a fail-safe maneuver is triggered, prompting the quadrotors to return safely to the charging station. In our three-quadrotor experiments, we observed a latency of $600\pm150$ ms, with $T_E$ set to 1.5 s.

All simulation and experimental code is publicly released. `RmeSch` is available as a Julia module that functions as a low-level filter for any high-level planner. We also provide a Python-ROS2 wrapper for Julia, a Docker container for easy deployment, and our in-house-developed Orin-based DevQuad platform D. Agrawal, Chen, and Panagou (2023).

# 9 Conclusion

This paper presented `mEclares`, a unified framework for adaptive ergodic exploration and robust energy-aware scheduling in persistent multi-robot missions. We addressed two key challenges in long-term autonomous operations: (i) planning informative trajectories in stochastic spatiotemporal environments, and (ii) coordinating energy-constrained robots through a shared mobile charging station. By modeling information decay using the clarity metric and integrating it into ergodic search, we enabled the construction of time-evolving target information distributions that guide exploration under uncertainty. To ensure task persistence, we introduced `RmeSch`, an online scheduling algorithm that guarantees mutually

exclusive access to the charging station and provides robustness to communication delays and central node failures via fail-safe coordination.

Our approach supports general nonlinear dynamics, handles uncertain charging station state, and scales to teams of robots. Through extensive simulations and real-world hardware experiments, we demonstrated the effectiveness of `mEclares` in maintaining persistent, informative coverage while adhering to energy and safety constraints. Theoretical guarantees further support the feasibility and robustness of our method under well-defined conditions.

Future work will explore extensions to fully decentralized scheduling under communication constraints, integration with online learning of environmental dynamics, and deployment in larger-scale, real-world missions with diverse robotic platforms.

# 10 Acknowledgments

# References

Abraham, I., Prabhakar, A., & Murphey, T. D. (2021). An ergodic measure for active learning from equilibrium. *IEEE Transactions on Automation Science and Engineering*, *18*(3), 917–931.

Agrawal, D., Chen, R., & Panagou, D. (2023). gatekeeper: Online safety verification and control for nonlinear systems in dynamic environments. In *2023 ieee/rsj international conference on intelligent robots and systems (iros)* (Vol. 40, pp. 259–266). doi: 10.1109/TRO.2024.3454415

Agrawal, D. R., & Panagou, D. (2023). Sensor-based planning and control for robotic systems: Introducing clarity and perceivability. *IEEE Control Systems Letters*, *7*, 2623–2628. doi: 10.1109/LCSYS.2023.3288493

Ames, A. D., Xu, X., Grizzle, J. W., & Tabuada, P. (2017). Control barrier function based

quadratic programs for safety critical systems. *IEEE Trans. on Automatic Control*, *62*(8), 3861–3876. doi: 10.1109/TAC.2016 .2638961

Asghar, A. B., Sundaram, S., & Smith, S. L. (2023). *Multi-robot persistent monitoring: Minimizing latency and number of robots with recharging constraints.*

Bentz, W., Hoang, T., Bayasgalan, E., & Panagou, D. (2018). Complete 3-d dynamic coverage in energy-constrained multi-uav sensor networks. *Autonomous Robots*, *42*, 825–851.

Bottarelli, L., Bicego, M., Blum, J., & Farinelli, A. (2019). Orienteering-based informative path planning for environmental monitoring. *Engineering Applications of Artificial Intelligence*, *77*, 46–58.

Chen, W., Khardon, R., & Liu, L. (2022, June). AK: Attentive Kernel for Information Gathering. In *Proceedings of robotics: Science and systems.* New York City, NY, USA. doi: 10.15607/RSS.2022.XVIII.047

Choset, H., & Pignon, P. (1998). Coverage path planning: The boustrophedon cellular decomposition. In *Field and service robotics* (pp. 203–209).

Coffin, H., Abraham, I., Sartoretti, G., Dillstrom, T., & Choset, H. (2022). Multi-agent dynamic ergodic search with low-information sensors. In *2022 international conference on robotics and automation (icra)* (pp. 11480–11486).

Couture-Beil, A., & Vaughan, R. T. (2009). Adaptive mobile charging stations for multi-robot systems. In *2009 ieee/rsj international conference on intelligent robots and systems* (pp. 1363–1368). doi: 10.1109/IROS.2009 .5354816

Dong, D., Berger, H., & Abraham, I. (2023). Time optimal ergodic search. *arXiv preprint arXiv:2305.11643*.

Dressel, L., & Kochenderfer, M. J. (2019). Tutorial on the generation of ergodic trajectories with projection-based gradient descent. *IET Cyper-Phys. Syst.: Theory & Appl.*, *4*(2), 89–100. Retrieved from https://doi.org/ 10.1049/iet-cps.2018.5032 doi: 10.1049/ iet-cps.2018.5032

Fouad, H., & Beltrame, G. (2022). Energy autonomy for robot systems with constrained resources. *IEEE Transactions on Robotics*, *38*(6), 3675–3693. doi: 10.1109/TRO.2022 .3175438

Gao, T., & Bhattacharya, S. (2019). Multi-robot charging strategies: A game-theoretic approach. *IEEE Robotics and Automation Letters*, *4*(3), 2823–2830. doi: 10.1109/LRA .2019.2921695

Garza, A. C. (2021). *Bayesian models for science-driven robotic exploration* (Unpublished doctoral dissertation). Carnegie Mellon University, Pittsburgh, PA.

Gawarkiewicz, G., Todd, R. E., Zhang, W., Partida, J., Gangopadhyay, A., Monim, M.-U.-H., . . . Dent, M. (2018). The changing nature of shelf-break exchange revealed by the ooi pioneer array. *Oceanography*, *31*(1), 60–70.

Gelb, A., et al. (1974). *Applied optimal estimation.* MIT press.

Jackson, B. E., Tracy, K., & Manchester, Z. (2021). Planning with attitude. *IEEE Robotics and Automation Letters*, *6*(3), 5658–5664.

Julian, K. D., & Kochenderfer, M. J. (2019). Distributed wildfire surveillance with autonomous aircraft using deep reinforcement learning. *Journal of Guidance, Control, and Dynamics*, *42*(8), 1768–1778.

Karapetyan, N., Asghar, A. B., Bhaskar, A., Shi, G., Manocha, D., & Tokekar, P. (2023). *Ag-cvg: Coverage planning with a mobile recharging ugv and an energy-constrained uav.*

Kenzin, M., Bychkov, I., & Maksimkin, N. (2020). 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, pp. 745–750). doi: 10.1109/CoDIT49905.2020 .9263974

Kingry, N., Liu, Y.-C., Martinez, M., Simon, B., Bang, Y., & Dai, R. (2017). 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)* (pp. 5233–5238). doi: 10.1109/IROS.2017.8206413

Li, B., Patankar, S., Moridian, B., & Mahmoudian, N. (2018). 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)* (pp. 1–8). doi: 10.1109/SSRR .2018.8468631

Lin, T. X., Yel, E., & Bezzo, N. (2018). Energy-aware persistent control of heterogeneous robotic systems. In *2018 annual american control conference (acc)* (pp. 2782–2787). doi: 10.23919/ACC.2018.8431238

Lin, X., Yazıcıoğlu, Y., & Aksaray, D. (2022). Robust planning for persistent surveillance with energy-constrained uavs and mobile charging stations. *IEEE Robotics and Automation Letters*, *7*(2), 4157–4164. doi: 10.1109/LRA.2022.3146938

Liu, L., & Michael, N. (2014). Energy-aware aerial vehicle deployment via bipartite graph matching. In *2014 international conference on unmanned aircraft systems (icuas)* (pp. 189–194). doi: 10.1109/ICUAS.2014 .6842255

Manjanna, S., Li, A. Q., Smith, R. N., Rekleitis, I., & Dudek, G. (2018). Heterogeneous multi-robot system for exploration and strategic water sampling. In *2018 ieee international conference on robotics and automation (icra)* (pp. 4873–4880).

Mathew, G., & Mezić, I. (2011). Metrics for ergodicity and design of ergodic dynamics for multi-agent systems. *Physica D: Nonlinear Phenomena*, *240*(4), 432–442. Retrieved from https :// www .sciencedirect .com / science / article / pii / S016727891000285X doi: https://doi.org/10.1016/j.physd.2010 .10.010

Mathew, N., Smith, S. L., & Waslander, S. L. (2015). Multirobot rendezvous planning for recharging in persistent tasks. *IEEE Transactions on Robotics*, *31*(1), 128–142. doi: 10.1109/TRO.2014.2380593

Mayer, S., Lischke, L., & Woźniak, P. W. (2019). Drones for search and rescue. In *1st international workshop on human-drone interaction*.

Meliou, A., Krause, A., Guestrin, C., & Hellerstein, J. M. (2007). Nonmyopic informative path planning in spatio-temporal models. In *Aaai* (Vol. 10, pp. 16–7).

Moon, B., Suvarna, N., Jong, A., Chatterjee, S., Yuan, J., & Scherer, S. (2025). Ia-tigris: An incremental and adaptive sampling-based planner for online informative path planning. *arXiv preprint arXiv:2502.15961*.

Naveed, K. B., Agrawal, D., Vermillion, C., & Panagou, D. (2024a). Eclares: Energy-aware clarity-driven ergodic search. In *2024 ieee international conference on robotics and automation (icra)* (pp. 14326–14332).

Naveed, K. B., Agrawal, D., Vermillion, C., & Panagou, D. (2024b). Eclares: Energy-aware clarity-driven ergodic search. In *2024 ieee international conference on robotics and automation (icra)* (pp. 14326–14332). doi: 10.1109/ICRA57147.2024.10611286

Naveed, K. B., Dang, A., Kumar, R., & Panagou, D. (2024). mesch: Multi-agent energy-aware scheduling for task persistence. *arXiv preprint arXiv:2406.04560*.

Notomista, G. (2022). Resilience and energy-awareness in constraint-driven-controlled multi-robot systems. In *2022 american control conference (acc)* (pp. 3682–3687). doi: 10.23919/ACC53348.2022.9867740

Notomista, G., Pacchierotti, C., & Giordano, P. R. (2022). Multi-robot persistent environmental monitoring based on constraint-driven execution of learned robot tasks. In *2022 international conference on robotics and automation (icra)* (pp. 6853–6859). doi: 10.1109/ICRA46639.2022.9811673

Notomista, G., Ruf, S. F., & Egerstedt, M. (2018). Persistification of robotic tasks using control barrier functions. *IEEE Robotics and Automation Letters*, *3*(2), 758–763. doi: 10.1109/LRA.2018.2789848

Rao, A., Breitfeld, A., Candela, A., Jensen, B., Wettergreen, D., & Choset, H. (2023). Multi-objective ergodic search for dynamic information maps. In *2023 ieee international conference on robotics and automation (icra)* (pp. 10197–10204). doi: 10.1109/ ICRA48891.2023.10160642

Seewald, A., Lerch, C. J., Chancán, M., Dollar, A. M., & Abraham, I. (2024). *Energy-aware ergodic search: Continuous exploration for multi-agent systems with battery constraints.*

Sujit, P., Sousa, J., & Pereira, F. L. (2009). Uav and auvs coordination for ocean exploration. In *Oceans 2009-europe* (pp. 1–7).

Sun, W., Sood, N., Dey, D., Ranade, G., Prakash,

S., & Kapoor, A. (2017). No-regret replanning under uncertainty. In *2017 ieee international conference on robotics and automation (icra)* (pp. 6420–6427).

Todd, R. E. (2020). Export of middle atlantic bight shelf waters near cape hatteras from two years of underwater glider observations. *Journal of Geophysical Research: Oceans*, *125*(4), e2019JC016006.

Waharte, S., & Trigoni, N. (2010). Supporting search and rescue operations with uavs. In *2010 international conference on emerging security technologies* (pp. 142–147). doi: 10 .1109/ICRA48891.2023.10161039

# Appendix A   Proofs

## A.1   Proof of Lemma 1

*Proof* To ensure safe and sequential return of all $N^*$ robots, the algorithm requires that each return is separated by at least $T_\delta$ seconds. The robot with the smallest remaining flight time, $T_{F,0}^{1'}$, is assumed to return first. Each subsequent robot must return with a delay of at least $T_\delta$ from the previous one.

Therefore, the last robot (i.e., the $N^*$-th robot) must complete its return no later than

$$T_R + (N^* - 1) \cdot T_\delta + T_e, \tag{A1}$$

where $T_R$ accounts for the time required to return, and $T_E$ accounts for a one-iteration delay before the return command can be issued.

To guarantee that even the last robot returns safely before depleting its energy, this total return time must be less than or equal to the smallest available remaining flight time:

$$T_R + (N^* - 1) \cdot T_\delta + T_E \leq T_{F,0}^{1'}. \tag{A2}$$

Rearranging the inequality:

$$(N^* - 1) \cdot T_\delta \leq T_{F,0}^{1'} - T_R - T_E, \tag{A3}$$

$$N^* - 1 \leq \frac{T_{F,0}^{1'} - T_R - T_E}{T_\delta}, \tag{A4}$$

$$N^* \leq 1 + \frac{T_{F,0}^{1'} - T_R - T_E}{T_\delta}. \tag{A5}$$

Taking the floor on the right-hand side ensures conservativeness and integer feasibility:

$$N^* = 1 + \left\lfloor \frac{T_{F,0}^{1'} - T_R - T_E}{T_\delta} \right\rfloor. \tag{A6}$$

$\square$

## A.2   Proof of Theorem 1

*Proof* We complete this proof considering two scenarios: In the first scenario, we prove recursive feasibility without central node failure. In the second scenario, we show that when the central node fails, the robots can be safely recovered while still respecting the constraints (23a) and (23b).

### A.2.1   Feasibility guarantee without central node failure

The proof, inspired by (D. Agrawal et al., 2023, Thm. 1), uses induction.

**Base Case**

At the time $t_1$ and iteration $j = 1$, since both Gap flag condition (35) and the Reserve SoC condition (38) are true, the candidate trajectories are committed for all rechargeable robots i.e. $\forall i \in \mathcal{R}$ and $\forall t_{m_1}^{i_1}, t_{m_2}^{i_2} \in \mathcal{T}$

$$x_1^{i,com}(t) \leftarrow x_1^{i,can}(t) \quad \forall t \in [t_1, t_{1,C}^i)$$

$$\implies \begin{cases} T_{F,1}^k > (T_R + T_E + kT_\delta) & \forall k \in \mathcal{R}' \\ e^i(t) > e_1^{res} & \forall t \in [t_1, t_{1,C}^i) \end{cases}$$

$$\implies \begin{cases} |t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta & \forall t \in [t_1, t_{1,R}) \\ e^i(t) > e_{min}^i & \forall t \in [t_1, t_{1,R}) \end{cases}$$

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 and $\forall t_{m_1}^{i_1}, t_{m_2}^{i_2} \in \mathcal{T}$

$$x_{j+1}^{i,com}(t) \leftarrow x_{j+1}^{i,can}(t) \quad \forall t \in [t_{j+1}, t_{j+1,C}^i)$$

$$\implies \begin{cases} T_{F,j+1}^k > (T_R + T_E + kT_\delta) & \forall k \in \mathcal{R}' \\ e^i(t) > e_{j+1}^{res} & \forall t \in [t_{j+1}, t_{j+1,C}^i) \end{cases}$$

$$\implies \begin{cases} |t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta & \forall t \in [t_{j+1}, t_{j+1,R}) \\ e^i(t) > e_{min}^i & \forall t \in [t_{j+1}, t_{j+1,R}) \end{cases}$$

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 $1'^{th}$ robot in $\mathcal{R}'$ returns either due to violation of Gap flag condition or the Reserve SoC condition, i.e.,

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

The candidate trajectories are committed for the remaining robots, i.e. $\forall k \in \mathcal{R}'\setminus\{1'\}$ and $\forall t_{m_1}^{i_1}, t_{m_2}^{i_2} \in \mathcal{T}$

$$x_{j+1}^{k,com} \leftarrow x_{j+1}^{k,can} \quad \forall t \in [t_{j+1}, t_{j+1,C}^k)$$

$$\implies \begin{cases} T_{F,j+1}^k > (T_R + T_E + kT_\delta) \\ e^k(t) > e_{j+1}^{res} \quad \forall t \in [t_{j+1}, t_{j+1,C}^k) \end{cases}$$

$$\implies \begin{cases} |t_{m_1}^{i_1} - t_{m_2}^{i_2}| > T_\delta \quad \forall t \in [t_{j+1}, t_{j+1,R}) \\ e^k(t) > e_{min}^k \qquad \forall t \in [t_{j+1}, t_{j+1,R}) \end{cases}$$

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

### A.2.2 Safe recovery and feasibility guarantee with central node failure

Suppose that at time $t_j$, all rechargeable robots generate new candidate trajectories and send requests to the central node for validation. Each robot then waits for a response until $t_{j-1,N}$, as specified by the fail-safe protocol. If no message is received from the central node by this deadline, each robot executes its onboard fail-safe maneuver using its previously committed trajectory $x_{j-1}^{i,com}$ and its stored return index $\text{ret}_{j-1}^i$.

- If $\text{ret}_{j-1}^i = 1$, the robot immediately continues following $x_{j-1}^{i,com}$, ensuring a return by $t_j + T_R$.
- If $\text{ret}_{j-1}^i > 1$, the robot idles for $\text{ret}_{j-1}^i T_\delta$ seconds and then executes a time-shifted version of $x_{j-1}^{i,com}$ over the interval $[t_j + \text{ret}_{j-1}^i T_\delta, t_j + \text{ret}_{j-1}^i T_\delta + T_R]$.

This structure ensures two properties:

1. **Gap constraint satisfaction:** The time-shifting mechanism guarantees that no two robots attempt to return simultaneously. Since each robot delays its return by $(\text{ret}_{j-1}^i - 1) \cdot T_\delta$, mutual exclusion at the charging station is preserved, and the gap condition (23b) holds.
2. **Minimum energy constraint satisfaction:** Since each robot had already committed a feasible trajectory at $t_{j-1}$ with enough energy to return after the assigned delay, the energy constraint (23a) remains satisfied.

Thus, even in the absence of centralized coordination, the robots return safely, respecting both the return gap and minimum energy requirements. Hence, recursive feasibility also holds under central node failure. □