Real-Time Area Coverage and Target Localization using Receding-Horizon Ergodic Exploration

Anastasia Mavrommati, Emmanouil Tzorakoleftherakis, Ian Abraham and Todd D. Murphey

Abstract—Although a number of solutions exist for the problems of coverage, search and target localization—commonly addressed separately—whether there exists a unified strategy that addresses these objectives in a coherent manner without being application-specific remains a largely open research question. In this paper, we develop a receding-horizon ergodic control approach, based on hybrid systems theory, that has the potential to fill this gap. The nonlinear model predictive control algorithm plans real-time motions that optimally improve ergodicity with respect to a distribution defined by the expected information density across the sensing domain. We establish a theoretical framework for global stability guarantees with respect to a distribution. Moreover, the approach is distributable across multiple agents, so that each agent can independently compute its own control while sharing statistics of its coverage across a communication network. We demonstrate the method in both simulation and in experiment in the context of target localization, illustrating that the algorithm is independent of the number of targets being tracked and can be run in real-time on computationally limited hardware platforms.

I. INTRODUCTION

This paper considers the problem of real-time motion planning for area search/coverage and target localization. Although the above operations are often considered separately, they essentially all share a common objective: tracking a specified distribution of information across the terrain. Our approach differs from common solutions of space grid decomposition in area coverage [1]–[5] and/or information maximization in target localization [6]–[13] by employing the metric of *ergodicity* to plan trajectories with spatial statistics that match the terrain spatial distribution in a continuous manner. By following this approach, we can establish a unified framework that achieves simultaneous search and localization of multiple targets (e.g., localizing detected targets while searching for new targets when the number of total targets is unknown) without added complexity. Previous work [14]–[16] has suggested using ergodic control for the purpose of motion planning for search and localization (albeit separately). However, due to its roots to optimal control theory, ergodic control has been associated with high computational cost that makes it impractical for real-time operations with varying information distribution.

Anastasia Mavrommati, Emmanouil Tzorakoleftherakis, Ian Abraham and Todd D. Murphey are with the Department of Mechanical Engineering, Northwestern University, 2145 Sheridan Road Evanston, IL 60208, USA Email: stacymav@u.northwestern.edu; man7therakis@u.northwestern.edu; ianabraham2015@u.northwestern.edu; t-murphey@northwestern.edu

The contribution of this work is a model predictive ergodic control algorithm that exhibits low execution times even for high-dimensional systems with complex nonlinear dynamics while providing stability guarantees over both dynamic states and information evolution. To the best of our knowledge, this paper includes the first application of an online ergodic control approach in real-time experimentation—here, using the sphero SPRK robot [17].

Ergodic theory relates the time-averaged behavior of a system to the set of all possible states of the system and is primarily used in the study of fluid mixing and communication. We use ergodicity to compare the statistics of a search trajectory to a terrain spatial distribution—the distribution may represent probability of detection for area search, regions of interest for area coverage and/or expected information density for target localization. To formulate the ergodic control algorithm, we employ hybrid systems theory to analytically compute the next control action that optimally improves ergodicity, in a receding-horizon manner. The algorithm—successfully applied to autonomous visual rendering in [18]—is shown to be particularly efficient in terms of execution time and capable of handling high-dimensional systems. The overall contribution of this paper combines the following features in a coherent manner.

Real-time execution:

In Sections VI-A and VI-B, we demonstrate in simulation real-time reactive control for a quadrotor model in SE(3). In Section VII, we show in experimentation how an ergodically controlled sphero SPRK robot can localize projected targets in real time using bearing-only measurements.

Adaptive performance: The proposed algorithm reactively adapts to a changing distribution of information across the workspace. We show how a UAV can adapt to new target estimates as well as to increasing numbers of targets.

Nonlinear agent dynamics: As opposed to geometric point-to-point approaches [4], [19], [20], the proposed algorithm controls agents with complex nonlinear dynamics, such as robotic fish [21], [22], 12-dimensional unmanned aerial vehicles (UAVs) [23], [24], etc., taking advantage of their dynamical features to achieve efficient area exploration.

Stability of information states: We establish requirements for ergodic stability of the closed-loop system resulting from the receding-horizon strategy in Section IV-B.

Multi-objective control capacity: The proposed algorithm can work in a shared control scenario by wrapping around controllers that implement other objectives. This dual control problem solution is achieved by encoding the non-information related (secondary mission) control signal as the nominal control input u_i^{nom} in the algorithm (see Algorithm 1). In the simulation examples, we show how this works by wrapping

the ergodic control algorithm around a PD controller for height regulation in order to control a quadrotor to explore a terrain.

Multi-agent distributability: We show how ergodic control is distributable to $N > 1$ agents with no added computational complexity. Each agent computes control actions *locally* using their own processing unit but still shares information *globally* with the other agents after each algorithm iteration.

Generalizability and robustness in multiple-targets localization: The complexity of sensor motion planning strategies usually scales with respect to the total number of targets, because a separate motion needs to be planned for each target [25], [26]. As opposed to this, the proposed ergodic control approach controls the robotic agents to track a non-parameterized information distribution across the terrain instead of individual targets independently, thus being completely decoupled from the estimation process and independent of the number of targets. Through a series of simulation examples and Monte Carlo experimental trials, we show that ergodically controlled agents with limited sensor ranges can reliably detect and localize static and moving targets in challenging situations where a) the total number of targets is unknown; b) a model of prior target behavior is not available; c) agents acquire bearing-only measurements; d) a standard Extended Kalman Filter (EKF) is used for bearing-only estimation.

Joint area coverage and target localization: Planning routes that simultaneously optimize the conflicting objectives of search and tracking is particularly challenging [27]–[29]. Here, we propose an ergodic control approach where the combined dual objective is encoded in the spatial information distribution that the statistics of the robotic agents trajectories must match.

II. Related Work

A. Area Search and Coverage

An area search function is required by many operations, including search-and-rescue [30], [31], hazard detection [20], agricultural spraying [32], solar radiation tracking [33] and monitoring of environmental phenomena, such as water quality in lakes [34]. In addition, complete area coverage navigation that requires the agent to pass through every region of the workspace [35] is an essential issue for cleaning robots [36], [37], autonomous underwater covering vehicles [38], [39], demining robots [40], automated harvesters [41], etc. Although slightly different in concept, both applications—search and coverage—involve motion planning for tracking a distribution of information across the terrain. For purposes of area search, this terrain spatial distribution indicates probability of detection and usually exhibits high variability in both space and time as it dynamically responds to new information about the target's state. In area coverage applications, on the other hand, the terrain distribution shows regions of interest and is normally near-uniform with possible occlusions.

A number of contributions in the area of robotic search and coverage decompose the exploration space to reduce problem complexity. Grid division methods of various geometries, such as Voronoi divisions [1]–[5], are commonly employed to accomplish this. While these methods work well, their scalability to more complex and larger terrains where

the number of discrete divisions increases, is a concern. In addition, existing methods plan paths that do not satisfy the robotic agents' dynamics and thus are not feasible system trajectories. This raises the need for an additional step where the path is treated as a series of waypoints and the agent is separately controlled to visit them all [4], [19], [20]. This double-step process—first, path planning and then, robot control—might result in hard-to-accomplish maneuvers for the robotic system. Finally, decomposition methods often do not respond well to dynamically changing environments—for example when probability of detection across the workspace changes in response to incoming data—because grid updates can be computationally intensive. Therefore, most existing solutions only perform non-adaptive path planning for search and coverage offline—i.e., when the distribution of information is known and constant[1]. As opposed to this, the algorithm described in this paper does not require decomposition of the workspace, by representing probability of detection as a continuous function over the terrain. Furthermore, it performs online motion planning by reactively responding to changes in the terrain spatial distribution in real time, while taking into account the agent's dynamics.

Multi-agent Coordinated Coverage:

The objective of multi-agent coverage control is to control the agents motion in order to collectively track a spatial probability-of-detection density function [43] across the terrain. Shared information is a necessary condition for coordination [44]. Several promising coverage control algorithms for mobile sensor networks have been proposed. In most cases, the objective is to control the agents to move to a static position that optimizes detection probability Φ, i.e., to compute and track the final states $x_\zeta(t_f)$ that maximize the sum of $\int \mathcal{S}(\|x - x_\zeta(t_f)\|)\Phi(x)dx$ over all agents ζ where $\mathcal{S}$ indicates sensor performance. Voronoi-based gradient descent [45], [46] is a popular approach but it can converge to local maxima. Other approaches employ cellular decomposition [47], simulated annealing [48], game theory [49], [50] and ergodic dynamics [51] to achieve distributed coverage. The main drawback is that existing algorithms do not consider time-dependent density functions Φ, so they are not suitable for realistic applications where probability of detection varies. Ergodic multi-agent coverage described in this paper differs from common coverage solutions in that it aims to control the agents so that the spatial statistics of their full trajectories—instead of solely their final states—optimize detection probability, i.e., the time-averaged integral $C(x) = \frac{1}{t_f - t_0} \int_{t_0}^{t_f} \delta[x - x_\zeta(t)]dt$, where δ is the Dirac delta, matches the spatial distribution $\Phi(x)$ as $t_f \to \infty$. This means that if we capture a single snapshot of the agents' ergodic motion, there is no guarantee that their current configuration will be maximizing the probability density. However, as time progresses the network of agents is bound to explore the terrain as extensively as possible by being ergodic with respect to the terrain distribution. An advantage of this approach compared to other coordinated coverage solutions, is that it can be performed online in order

---

[1]To overcome this issue when monitoring environments with changing distributions, an alternative solution is to control only the speed of the robotic agents over a predefined path [42].

to cover terrains with time-dependent (or sensed in real time) density functions.

B. Motion Planning for Target Localization

Target localization[2] refers to the process of acquiring and using sensor measurements to estimate the state of a single or multiple targets. One of the main challenges involved with target localization is developing a sensor trajectory such that high information measurements are acquired. To achieve this, some methods perform information maximization (IM) [6]–[12], [52] usually by compressing an information metric (such as Fisher Information Matrix (FIM) [24], [53] and Bayesian utility functions [10]) to a scalar cost function (e.g., using the determinant [12], trace [11], or eigenvalue [54], [55] of the information matrix) and then generating trajectories that optimize this cost. The most general approaches to solving these problems involve numerical techniques [56], classical optimal control theory [6], and dynamic programming [34], [57], which tend to be either computationally intensive or application specific (e.g., consider only static and/or constant velocity targets). Compared to IM techniques [6]–[12], [52], the proposed algorithm explores an expected information density map by optimally improving ergodicity, thus being more time-efficient and more robust to the existence of local maxima in information. In addition, it allows tracking multiple moving targets without added complexity as opposed to most IM techniques that would need to plan a motion for each target separately. To overcome this issue with IM techniques, Dames et al. [58], [59] propose an estimation filter that estimates the targets' density—instead of individual labeled targets—over time, thus rendering IM complexity independent of the number of targets. However, the proposed trajectory generation methodology relies on exhaustive search requiring discretization of the controls space.

Because IM techniques tend to exhibit prohibitive execution times for moving targets, alternative methods of diverse nature have been proposed for use in real-world applications. A non-continuous grid decomposition strategy for planning parameterized paths for UAVs is proposed in [60] with the objective to localize a single target by maximizing the probability of detection when the target motion is modeled as a Markov process. Standoff tracking techniques are commonly used to control the agent to achieve a desired standoff configuration from the target usually by orbiting around it [26], [61], [62]. A probabilistic planning approach for localizing a group of targets using vision sensors is detailed in [63]. In [64], a UAV is used to track a target in constant wind considering control input constrains, but the planned path is predefined to converge to a desired circular orbit around the target. A rule-based guidance strategy for localizing moving targets is introduced in [65]. In [66], the problem of real-time path planning for tracking a single target while avoiding obstacles is addressed through a combination of methodologies. In general, as opposed to ergodic exploration, the above approaches focus

on and are only applicable in special real-world situations and do not generalize directly to general multiple-target tracking situations. For a complete and extensive comparison of ergodic localization to other motion planning approaches, the reader is referred to [14].

In this paper, we use as an example the problem of bearing-only localization. Many real-world systems use angle-only sensors for target localization, such as submarines with passive sonar, mobile robots with directional radio antenna [67], and unmanned aerial vehicles (UAVs) using images from an optical sensor [68]. Bearing-only systems require some amount of maneuver to measure range with minimum uncertainty [6]. The majority of existing solutions for UAV bearings-only target motion planning, produce circular trajectories above the target's estimated position [23], [69]. However, this solution is only viable if there is low uncertainty over the target's position. In addition, if the target is moving, the operator may not know what the best vehicle path is. In this paper, this drawback is overcome by exploring an expected information density that is updated in real time while the targets are moving based on the current targets' estimate.

Cooperative Target Localization: The greatest body of work in the area of cooperative target localization is comprised by standoff tracking techniques. Vector fields [26], nonlinear feedback [70] and nonlinear model predictive control [61] are some of the control methodologies that have been used for achieving the desired standoff configuration for a target. The motion of the robotic agents is coordinated in a geometrical manner: two robotic agents orbit the target at a nominal stand-off distance and maintain a specified angular separation; when more agents are considered, they are controlled to achieve a uniform angular separation on a circle around the target. A dynamic programming technique that minimizes geolocation error covariance is proposed in [71]. However, the solution is not generalizable to multiple robotic agents and targets. The robots are controlled to seek informative observations by moving along the gradient of mutual information in [52]. The main concern in using these strategies is scalability to multiple targets, as the robots are separately controlled to fly around each single target [72]. To overcome this issue, we propose an algorithm that tracks a non-parameterized information density across the terrain instead of individual targets independently, thus being completely decoupled from the estimation process and the number of targets.

C. Ergodic Control Algorithms

There are a few other algorithms that perform ergodic control in the sense that they optimize the ergodicity metric in (5). Mathew and Mezić in [73] derive analytic ergodic control formulas for simple linear dynamics (single and double integrator) by minimizing the Hamiltonian [74] in the limit as the receding time horizon goes to zero. Although closed-form, their solution is not generalizable to arbitrary nonlinear system dynamics and it also augments the state vector to include the coefficients difference so that the final system dimensionality is nominally infinite. Miller et al. [75] propose an open-loop trajectory optimization technique using a finite-time horizon.

---

[2]While often "localization" refers to static targets and "tracking" is used for moving targets, here for consistency we will use the term "localization" to describe estimation of both static and moving targets.

TABLE I
LIST OF VARIABLES

Symbol	Description	Symbol	Description
x	system dynamic states	$J_{\mathcal{E}}$	receding-horizon ergodic cost
u	system inputs	t_0^{erg}	initial time of ergodic exploration
n	number of dynamic states	t_s	algorithm sampling time
m	number of system inputs	Q	weight on ergodic cost
ν	number of ergodically explored states ($\nu \leq n$)	R	weight on control effort
L_i	i-th dimension of search domain	$\rho_{\mathcal{E}}$	ergodic costate/adjoint
$\Phi(\cdot)$	spatial distribution on the search domain	α_d	desired rate of change
ϕ_k	Fourier coefficients of $\Phi(\cdot)$	u^{nom}	nominal control
c_k^i	Parameterized trajectory spatial statistics at time step t_i	$u_i^*(t), x_i^*(t)$	open-loop control and state trajectories at time step t_i
K	highest order of Fourier coefficients	$u^{def}(t), x^{def}(t)$	default control and resulting state trajectory
k	set of ν coefficient indices $\{k_1, k_2, ..., k_\nu\}$	$\bar{x}_{cl}$	closed-loop state trajectory
F_k	k-th Fourier basis function	u_A, λ_A, τ_A	magnitude, duration and application time of action A
T	open-loop time horizon	$C_{\mathcal{E}}$	cost contractive constraint
u_s^*	schedule of candidate infinitesimal actions	M_{erg}	ergodic memory
N	number of agents on the field	ζ	agents performing exploration
M	number of single-target coordinates	α	single-target parameters to be estimated
z	sensor measurement	μ	number of sensor measured quantities
$\Upsilon(\cdot)$	deterministic sensor measurement model	$\mathcal{I}$	Fisher Information matrix
Σ	covariance of measurement model	$\boldsymbol{\Phi}(\cdot)$	expected information matrix
$\mathcal{F}(\cdot)$	target dynamics		

This algorithm is ideal for generating optimal ergodic solutions with a prescribed time window. However, it exhibits relatively high computational cost that does not favor real-time algorithm application in a receding-horizon format. This approach has been used for offline receding-horizon exploration of unknown environments [15] and localization of a single static target [14], [16] in cases where real-time control is not imperative. De La Torre et al. [76] propose a stochastic differential dynamic programming algorithm for ergodic exploration in the presence of stochastic sensor dynamics. The ergodic control algorithm in this paper differs from these existing approaches in that it employs hybrid systems theory to perform ergodic exploration fast, in real time, while adaptively responding to changes in the information distribution as required for tracking moving targets.

III. ERGODICITY METRIC

For area coverage and target localization using ergodic theory, the objective is to control an agent so that the amount of time spent in any given area of a specified search domain is proportional to the integral of a spatial distribution over that same domain. This section describes an ergodicity metric that satisfies this objective.

Consider a search domain that is a bounded ν-dimensional workspace $\mathcal{X}_\nu \subset \mathbb{R}^\nu$ defined as $[0, L_1] \times [0, L_2] \times ... \times [0, L_\nu]$ with $\nu \leq n$, where n is the total number of the system dynamic states. If $s \in \mathcal{X}^\nu$ denotes a point in the search domain, the spatial distribution over the search domain is denoted as $\Phi(s) : \mathcal{X}^\nu \to \mathbb{R}$, and it can represent probability of detection in search area coverage operations, such as search-and-rescue,

surveillance, inspection etc. or expected information density in target localization tasks as in Section V. The spatial statistics of a trajectory $x_\nu(t)$ are quantified by the percentage of time spent in each region of the workspace as

$$C(s) = \frac{1}{T} \int_{t_0}^{t_0+T} \delta[s - x_\nu(t)]dt \tag{1}$$

where δ is the Dirac delta. We use the distance from ergodicity between the spatial statistics of the time-averaged trajectory and the terrain spatial distribution as a metric. To drive the spatial statistics of a trajectory $x_\nu(t)$ to match those of the distribution $\Phi(s)$, we need to choose a norm on the difference between the distributions $\Phi(s)$ and $C(s)$. As in [14], we quantify the difference between the distributions, i.e., the *distance from ergodicity*, using the sum of the weighted squared distance between the Fourier coefficients ϕ_k of $\Phi(s)$, and the coefficients c_k of the distribution $C(s)$ representing the time-averaged trajectory. In particular, the Fourier coefficients ϕ_k and c_k are calculated respectively as

$$\phi_k = \int_{\mathcal{X}^\nu} \Phi(x_\nu)F_k(x_\nu)dx_\nu \tag{2}$$

and

$$c_k = \frac{1}{T} \int_{t_0}^{t_0+T} F_k(x_\nu(t))dt \tag{3}$$

where F_k is a Fourier basis function, as derived in [73], and T is the open-loop time horizon. Here, we use the following

choice of basis function:

$$F_k(s) = \frac{1}{h_k} \prod_{i=1}^{\nu} \cos\left(\frac{k_i\pi}{L_i} s_i\right), \qquad (4)$$

where $k \in \mathcal{K}$ is a set of ν coefficient indices $\{k_1, k_2, ..., k_\nu\}$ with $k_i \in \mathbb{N}$ so that $\mathcal{K} = \{k \in \mathbb{N}^\nu : 0 \leq k_i \leq K\}$, $K \in \mathbb{N}$ is the highest order of coefficients calculated along each of the ν dimensions, and h_k is a normalizing factor [73]. It should be noted, however, that any set of basis functions that is differentiable in the state and can be evaluated along the trajectory can be used in the derivation of the ergodic metric. Using the above, the ergodic metric on $x_\nu \in \mathcal{X}^\nu$ is defined as in [14], [73], [75]

$$\mathcal{E}(x_\nu(t)) = \sum_{k \in \mathcal{K}} \Lambda_k [c_k(x_\nu(t)) - \phi_k]^2 \qquad (5)$$

with $\Lambda_k = \frac{1}{(1+\|k\|^2)^s}$ and $s = \frac{\nu+1}{2}$, which places larger weight on lower frequency information so that when $K \to \infty$ the series converges.

IV. Receding-horizon ergodic exploration

A. Algorithm Derivation

We shall consider nonlinear control affine systems with input constraints such that

$$\dot{x} = f(t, x, u) = g(t, x) + h(t, x)u \quad \forall t \qquad (6)$$

with $u \in \mathcal{U}$ and

$$\mathcal{U} := \left\{ u \in \mathbb{R}^m : u_{min} \leq u \leq u_{max}, \; u_{min} < 0 < u_{max} \right\},$$

i.e., systems that can be nonlinear with respect to the state vector, $x : \mathbb{R} \to \mathcal{X}$, but are assumed to be linear (or linearized) with respect to the control vector, $u : \mathbb{R} \to \mathcal{U}$. The state will sometimes be denoted as $t \mapsto x(t; x(t_i), u(\cdot))$ when we want to make explicit the dependence on the initial state (and time), and corresponding control signal. Using the metric (5), receding-horizon ergodic control must optimally improve the following cost at each time step t_i:

$$J_\mathcal{E} = Q \sum_{k \in \mathcal{K}} \Lambda_k \Bigg[\underbrace{\frac{1}{t_i + T - t_0^{erg}} \int_{t_0^{erg}}^{t_i+T} F_k(x(t))dt - \phi_k}_{c_k^i} \Bigg]^2 \qquad (7)$$

where t_0^{erg} is the user-defined initial time of ergodic exploration, $x \in \mathcal{X}^\nu$ and $Q \in \mathbb{R}$ weights the ergodic cost against control effort weighted by R in (13). Henceforth, for brevity we refer to the trajectory of the set of states to be ergodically explored as $x(t)$ instead of $x_\nu(t)$, although it should be clear that the ergodically explored states might or might not be all the states of the system dynamics (i.e., $\nu \leq n$).

To understand the challenges of optimizing (7), we distinguish between the dynamic states of the controlled system, $x \in \mathbb{R}^n$—e.g., the 12 states denoting position and heading, and their velocities, in quadrotor dynamics—and the information states $c_k(x(\cdot))$ in (3), i.e., the parameterized time-averaged statistics of the followed trajectory over a finite time duration. The main difficulty in optimizing ergodicity is that the ergodic cost functional in (7) is non-quadratic and does not follow

the Bolza form—consisting of a running and terminal cost [77]—with respect to the dynamic states. To address this, infinite-dimensional trajectory optimization methods that are independent of the cost functional form have been employed [14] to optimize ergodicity. However, the computational cost of such iterative methods is prohibitive for real-time control in a receding-horizon approach. Another method involves change of coordinates so that the cost functional is rendered quadratic with respect to the information states parameterized by Fourier coefficients. This allows the use of traditional optimal control approaches e.g., LQR, DDP, SQP etc. (see for example [76]). However, this approach entails optimization over an extended set of states (the number of parameterized information states is usually significantly larger than the dynamic states) which inhibits real-time execution. In addition, and perhaps more importantly, defining a running cost on the information states results in unnecessarily repetitive integration of the dynamic state trajectories. To avoid this, an option would be to optimize a terminal cost only, but this proves problematic in establishing stability of Model Predictive Control (MPC) algorithms (see [78]).

To overcome the aforementioned issues, we seek to formulate an algorithm that a) computes control actions that guarantee contraction of the ergodic cost at each time step b) naturally uses current sensor feedback to compute controls fast, in real time. For these reasons, we choose to frame the control problem as a hybrid control problem, similarly to Sequential Action Control (SAC) in [79], [80]. By doing this, we are able to formulate an ergodic control algorithm that is rendered fast enough for real time operation—as opposed to traditional model predictive control algorithms that are usually computationally expensive [81]—for two main reasons: a) a single control action is calculated at every time step using a closed-form algebraic expression and b) this control action aims to optimally improve ergodicity (instead of optimizing it) by an amount that guarantees stability with respect to x and c_i.

An overview of the algorithm is given in Algorithm 1. Once the Fourier coefficients ϕ_k of the spatial distribution $\Phi(x)$ have been calculated, the algorithm follows a receding-horizon approach; controls are obtained by repeatedly solving online an open-loop ergodic control problem $\mathcal{P}_\mathcal{E}$ every t_s seconds (with sampling frequency $1/t_s$), every time using the current measure of the system dynamic state x. The following definitions are necessary before introducing the open-loop problem.

Definition 1: An action A is defined by the triplet consisting of a control's value, $u_A \in \mathcal{U}$, application duration, $\lambda_A \in \mathbb{R}^+$ and application time, $\tau_A \in \mathbb{R}$, such that $A := \{u_A, \lambda_A, \tau_A\}$.

Definition 2: Nominal control $u^{nom} : \mathbb{R} \to \mathcal{U}$, provides a nominal trajectory around which the algorithm provides feedback. When applying ergodic control as a standalone controller, $u^{nom}(\cdot)$ is either zero or constant. Alternatively, $u^{nom}(\cdot)$ may be an optimized feedforward or state-feedback controller.

The open-loop problem $\mathcal{P}_\mathcal{E}$ that is solved in each iteration of

Algorithm 1 Receding-horizon ergodic exploration (RHEE)

Inputs: initial time t_0, initial state x_0, terrain spatial distribution $\Phi(x)$, ergodic initial time t_0^{erg}, final time t_f

Output: closed-loop ergodic trajectory $\bar{x}_{cl} : [t_0, t_f] \to \mathcal{X}$

Define ergodic cost weight Q, highest order of coefficients K, control weight R, search domain bounds $\{L_1, ..., L_v\}$, sampling time t_s, desired rate of change α_d, time horizon T.

Initialize nominal control u^{nom}, step $i = 0$.

- ■ Calculate ϕ_k using (2).
- ■ While $t_i < t_f$
  - □ Solve open-loop problem $\mathcal{P}_{\mathcal{E}}(t_i, x_i, T)$ to get u_i^*:
    1) Simulate system (6) for $t \in [t_i, t_i + T]$ under u_i^{def} to get $x(t)$.
    2) Simulate (12) for $t \in [t_i, t_i + T]$.
    3) Compute u_s^* using (14).
    4) Determine action application time t_A and value u_A by minimizing (15).
    5) Determine action duration λ_A using the line search process in Section IV-A4 and the condition in (9) with $C_{\mathcal{E}}$ in (19).
  - □ Apply u_i^* to (6) for $t \in [t_i, t_i + t_s]$ to get $\bar{x}_{cl} \forall t \in [t_i, t_i + t_s]$.
  - □ Define $t_{i+1} = t_i + t_s$, $x_{i+1} = \bar{x}_{cl}(t_{i+1})$.
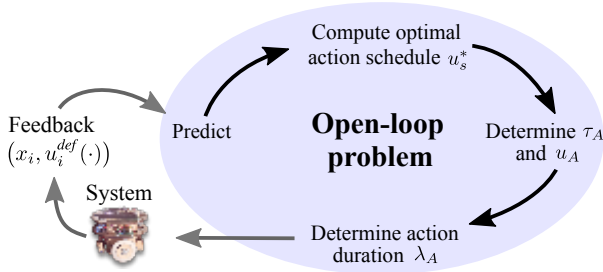 - □ $i \leftarrow i + 1$

  end while

Fig. 1. An overview of the ergodic control process. One major difference between the proposed ergodic control algorithm and traditional MPC approaches is that the open-loop problem can be solved without employing nonlinear programming solvers [82] by using hybrid systems theory. In order to solve (8), the algorithm follows four steps as illustrated above.

the receding horizon strategy can now be defined as follows[3].

$$\mathcal{P}_{\mathcal{E}}(t_i, x_i, T) : \tag{8}$$

Find action A such that

$$J_{\mathcal{E}}(x(t; x_i, u_i^*(\cdot))) - J_{\mathcal{E}}(x(t; x_i, u_i^{def}(\cdot))) < C_{\mathcal{E}} \tag{9}$$

subject to

$$u_i^*(t) = \begin{cases} u_A & \tau_A \leq t \leq \tau_A + \lambda_A \\ u_i^{def}(t) & \text{else} \end{cases},$$

and (6) with $t \in [t_i, t_i + T]$ and $x(t_i) = x_i$.

where $C_{\mathcal{E}}$ is a quantity that guarantees stability (see Section IV-B) and u_i^{def} and x_i^{def} are defined below.

Definition 3: Default control $u_i^{def} : [t_i, t_i + T] \to \mathcal{U}$, is de-

---

[3]From now on, subscript i will denote the i-th time step, starting from $i = 0$.

fined as

$$u_i^{def}(t) = \begin{cases} u_{i-1}^*(t) & t_i \leq t \leq t_i + T - t_s \\ u_i^{nom}(t) & t_i + T - t_s < t \leq t_i + T \end{cases}, \tag{10}$$

with $u_0^{def}(\cdot) \equiv u_0^{nom}(\cdot)$, $u_{i-1}^* : [t_{i-1}, t_{i-1} + T] \to \mathcal{U}$ the output of $\mathcal{P}_{\mathcal{E}}(t_{i-1}, x_{i-1}, T)$ from the previous time step $i-1$—corresponding to $x_{i-1}^*(\cdot)$—and $t_s = t_i - t_{i-1}$ the sampling period (Fig. 1). The system trajectory corresponding to application of default control will be denoted as $x(t; x(t_i), u^{def}(\cdot))$ or $x^{def}(\cdot)$ for brevity.

The structure of $u_i^*(t)$ in (8) is a design choice that allows us to compute a single control action A, since $u_i^{def}(t)$ is known at each time step t_i. As pointed out in the second paragraph of Section IV-A, this single-action control problem renders the algorithm fast enough for real-time execution. Moreover, the structure of default control $u_i^{def}(t)$ in (10) indicates that the algorithm stores actions calculated in the previous time steps (included in $u_{i-1}^*(t)$). By defining default control this way, and thus storing past actions, we are able to rewrite (9) as a constraint on sequential open-loop costs, as in (20), in order to show stability.

The following proposition is necessary before going though the steps for solving $\mathcal{P}_{\mathcal{E}}$.

Proposition 1: Consider the case where the system (6) evolves according to default control dynamics $f(t, x(t), u_i^{def}(t))$, and action u_A is applied at time τ (dynamics switch to $f(t, x(t), u_A)$) for an infinitesimal duration $\lambda \to 0$ before switching back to default control. In this case, the ergodic mode insertion gradient $\frac{\partial J_{\mathcal{E}}}{\partial \lambda}$ evaluated at $t = \tau$ measures the first-order sensitivity of the ergodic cost (7) to infinitesimal application of control action u_A and is calculated as

$$\left. \frac{\partial J_{\mathcal{E}}}{\partial \lambda} \right|_{\tau} = \rho_{\mathcal{E}}(\tau) \left[f(\tau, x_i^{def}(\tau), u_A) - f(\tau, x_i^{def}(\tau), u_i^{def}(\tau)) \right] \tag{11}$$

with

$$\dot{\rho}_{\mathcal{E}} = -\ell(t, x_i^{def})^T - D_x f(t, x_i^{def}, u_i^{def})^T \cdot \rho_{\mathcal{E}} \tag{12}$$
$$\text{subject to } \rho_{\mathcal{E}}(t_i + T) = \mathbf{0}$$

and $\ell(t, x) = \frac{2Q}{t_i + T - t_0^{erg}} \sum_{k \in \mathcal{K}} \left\{ \Lambda_k [c_k^i - \phi_k] \frac{\partial F_k(x(t))}{\partial x(t)} \right\}$.

Proof The proof is provided in Appendix A.

Note that the open loop problem $\mathcal{P}_{\mathcal{E}}$ in (8) only requires that the cost $J_{\mathcal{E}}$ is decreased by a specific amount $C_{\mathcal{E}}$ (at minimum), instead of maximizing the cost contraction. This allows us to quickly compute a solution, while still achieving stability, following the steps listed in the following.

1) Predict: In this step, the algorithm evaluates the system (6) from the current state x_i and time t_i, with $u_i^{def}(t)$ for $t \in [t_i, t_i + T]$. In addition, it uses the predicted state trajectory to backward simulate $\rho_{\mathcal{E}}$ that satisfies (12).

2) Compute optimal action schedule $u_s^*(\cdot)$: In this step, we compute a schedule $u_s^* : [t_i, t_i + T] \to \mathbb{R}^m$ which contains candidate infinitesimal actions. Specifically, $u_s^*(\cdot)$ contains candidate action values and their corresponding application times, but assumes $\lambda \to 0^+$ for all. The final u_A and τ_A will be selected from these candidates in step three of the solution

process such that $u_A = u_s^*(\tau_A)$, while a finite duration λ_A will be selected in the final step. The optimal action schedule $u_s^*(\cdot)$ is calculated by minimizing

$$J_{u_s} = \frac{1}{2} \int_{t_i}^{t_i+T} \left[\frac{\partial J_\mathcal{E}}{\partial \lambda}(t) - \alpha_d \right]^2 + \|u_s(t)\|_R^2 \, dt, \quad (13)$$

$$\frac{\partial J_\mathcal{E}}{\partial \lambda}(t) = \rho_\mathcal{E}(t)^T \left[f(t, x_i^{def}(t), u_s(t)) - f(t, x_i^{def}(t), u_i^{def}(t)) \right]$$

where the quantity $\frac{\partial J_\mathcal{E}}{\partial \lambda}(\cdot)$ (see Proposition 1), called the mode insertion gradient [83], denotes the rate of change of the cost with respect to a switch of infinitesimal duration λ in the dynamics of the system. In this case, $\frac{dJ_\mathcal{E}}{d\lambda}(\cdot)$ shows how the cost will change if we introduce a single infinitesimal switch from $f(t, x_i^{def}(t), u_i^{def}(t))$ to $f(t, x_i^{def}(t), u_s(t))$ at some point in the time window $[t_i, t_i + T]$. The parameter $\alpha_d \in \mathbb{R}^-$ is user specified and allows the designer to influence how aggressively each action value in the schedule $u_s^*(t)$ improves the cost.

Based on the evaluation of the dynamics (6), and (12) completed in the prediction step (Section IV-A1), minimization of (13) leads to the following closed-form expression for the optimal action schedule:

$$u_s^*(t) = (\Lambda + R^T)^{-1} \left[\Lambda u_i^{def}(t) + h(t, x_i^{def}(t))^T \rho_\mathcal{E}(t) \alpha_d \right], \quad (14)$$

where $\Lambda \triangleq h(t, x_i^{def}(t))^T \rho_\mathcal{E}(t)\rho_\mathcal{E}(t)^T h(t, x_i^{def}(t))$. The infinitesimal action schedule can then be directly saturated to satisfy any min/max control constraints of the form $u_{min,k} < 0 < u_{max,k} \; \forall k \in \{1, \dots, m\}$ such that $u_s^* \in \mathcal{U}$ without additional computational overhead (see [79] for proof).

3) Determine application time τ_A (and thus u_A value): Recall that the curve $u_s^*(\cdot)$ provides the values and application times of possible infinitesimal actions that the algorithm could take at different times to optimally improve system performance from that time. In this step the algorithm chooses one of these actions to apply, i.e., chooses the application time τ_A and thus an action value u_A such that $u_A = u_s^*(\tau_A)$. To do that, $u_s^*(\cdot)$ is searched for a time τ_A that minimizes

$$J_t(\tau) = \frac{\partial J_\mathcal{E}}{\partial \lambda}\bigg|_\tau, \quad (15)$$

$$\frac{\partial J_\mathcal{E}}{\partial \lambda}\bigg|_\tau = \rho_\mathcal{E}(\tau)^T \left[f(\tau, x_i^{def}(\tau), u_s^*(\tau)) - f(\tau, x_i^{def}(\tau), u_i^{def}(\tau)) \right]$$

$$\text{subject to } \tau \in [t_i, t_i + T].$$

Notice that the cost (15) is actually the ergodic mode insertion gradient evaluated at the optimal schedule $u_s^*(\cdot)$. Thus, minimization of (15) is equivalent to selecting the infinitesimal action from $u_s^*(\cdot)$ that will generate the greatest cost reduction relative to only applying default control.

4) Determine control duration λ_A: The final step in synthesizing an ergodic control action is to choose how long to act, i.e., a finite control duration λ_A, such that condition (9) is satisfied. From [83], [84], there is a non-zero neighborhood around $\lambda \to 0^+$ where the mode insertion gradient models the change in cost in (9) to first order, and thus, a finite duration λ_A exists that guarantees descent. In particular, for *finite* durations

λ in this neighborhood we can write

$$J_\mathcal{E}(x(t; x_i, u_i^*(\cdot))) - J_\mathcal{E}(x(t; x_i, u_i^{nom}(\cdot)))$$

$$= \Delta J_\mathcal{E} \approx \frac{\partial J_\mathcal{E}}{\partial \lambda}\bigg|_{\tau_A} \lambda. \quad (16)$$

Then, a finite action duration λ_A can be calculated by employing a *line search* process [84].

After computing the duration λ_A, the control action A is fully specified (it has a value, an application time and a duration) and thus the solution $u_i^*(t)$ of problem $\mathcal{P}_\mathcal{E}$ has been determined. By iterating on this process (Section IV-A1 until Section IV-A4), we rapidly synthesize piecewise continuous, input-constrained ergodic control laws for nonlinear systems.

Algorithm 2 Reactive RHEE for varying $\Phi(x)$

Define ergodic memory M_{erg}, distribution sampling time t_ϕ.
Initialize current time t_{curr}, current state x_{curr}.

While $t_{curr} < \infty$
  1) Receive/compute current $\Phi_{curr}(x)$.
  2) $t_0^{erg} \leftarrow t_{curr} - M_{erg}$
  3) $t_{final} \leftarrow t_{curr} + t_\phi$
  4) $\bar{x}_{cl} = RHEE(t_{curr}, x_{curr}, t_0^{erg}, t_{final}, \Phi_{curr}(x))$
  5) $t_{curr} \leftarrow t_{curr} + t_\phi$
  6) $x_{curr} = \bar{x}_{cl}(t_{final})$
end while

The Receding-Horizon Ergodic Exploration (RHEE) process for a dynamically varying $\Phi(x)$ is given in Algorithm 2. The re-initialization of RHEE when a new $\Phi(x)$ is available serves two purposes: first, it allows for the new coefficients ϕ_k to be calculated[4] and second, it allows the update of the ergodic initial time t_0^{erg}.

The ergodic initial time t_0^{erg} is particularly important for the algorithm performance because it regulates how far back in the past the algorithm should look when comparing the spatial statistics of the trajectory (parameterized by c_k) to the input spatial distribution (parameterized by ϕ_k). If the spatial distribution is regularly changing to incorporate new data (for example in the case that the distribution represents expected information density in target localization as we will see in Section V), it is undesirable for the algorithm to use state trajectory data all the way since the beginning of exploration. At the same time, "recently" visited states must be known to the algorithm so that it avoids visiting them multiple times during a short time frame. To specify our notion of "recently" depending on the application, we use the parameter M_{erg} in Algorithm 2 which we call "ergodic memory" and simply indicates how many time units in the past the algorithm has access to, so that it is $t_0^{erg} = t_0 - M_{erg}$ every time RHEE is re-initialized at time t_0.

The C++ code for both Algorithms 1 and 2 is available online at github.com/MurpheyLab.

---

[4]This corresponds to the general case when the distribution time evolution is unknown. If, however, $\Phi(x)$ is a time-varying distribution with known evolution in time, we can pre-calculate the coefficients ϕ_k offline to reduce the computational cost further.

a) Remarks on computing c_k^i: The cost $J_\mathcal{E}$ in (7) depends on the full state trajectory from a defined initial time $t = t_0^{erg} \le t_i$ in the past (instead of t_i as in common tracking objectives) to $t = t_i + T$, which could arise concerns with regard to execution time and computational cost. Here, we show how to compute c_k^i in a way that avoids integration over an infinitely increasing time duration as $t_i \to \infty$. To calculate trajectory coefficients c_k^i at time step t_i with $k \in \mathcal{K}$, and thus cost $J_\mathcal{E}$, notice that:

$$c_k^i = \frac{1}{t_i + T - t_0^{erg}} \int_{t_0^{erg}}^{t_i+T} F_k(x(t))dt = \tag{17}$$

$$= \underbrace{\frac{1}{t_i + T - t_0^{erg}} \int_{t_0^{erg}}^{t_i} F_k(x(t))dt}_{\bar{c}_k^{(i)}} + \frac{1}{t_i + T - t_0^{erg}} \int_{t_i}^{t_i+T} F_k(x(t))dt$$

where recursively

$$\bar{c}_k^{(i)} = \frac{t_{i-1} + T - t_0^{erg}}{t_i + T - t_0^{erg}} \bar{c}_k^{(i-1)} + \frac{1}{t_i + T - t_0^{erg}} \int_{t_{i-1}}^{t_i} F_k(x(t))dt$$

$\forall \ i \ge 1, \ k \in \mathcal{K}$ with $\bar{c}_k^{(0)} = 0$.

Therefore, only the current open-loop trajectory $x(t)$ for all $t \in [t_i, t_i + T]$ and a set of $(K+1)^\nu$ coefficients $\bar{c}_k^{(i)} \in \mathbb{R}$ are needed for calculation of c_k^i and thus $J_\mathcal{E}$ at the ith time step. Coefficients $\bar{c}_k^{(i)} \in \mathbb{R}$ can be updated at the end of the ith time step and stored for use in the next time step at t_{i+1}. This provides the advantage that although the cost depends on an integral over time with an increasing upper limit, the amount of stored data needed for cost calculation does not increase but remains constant as time progresses.

B. Stability analysis

In this section, we establish the requirements for ergodic stability of the closed-loop system resulting from the receding-horizon strategy in Algorithm 1. To achieve closed-loop stability for Algorithm 1, we apply a contractive constraint [85]–[88] on the cost. For this reason, we define $C_\mathcal{E}$ from the open-loop problem (8) as follows.

Definition 4: Let Q be the set of trajectories $x(\cdot) : \mathbb{R} \to \mathcal{X}$ in (6) and $Q_d \subset Q$ the subset that satisfies $c_k(x(\cdot)) - \phi_k = 0$ for all k. Suppose $L(x(\cdot), u, t) : Q \times \mathbb{R}^m \times \mathbb{R} \to \mathbb{R}$ is defined as follows:

$$L(x(\cdot), u, t) := \frac{2Q}{t - t_0^{erg}} \sum_k \left\{ \Lambda_k \left[c_k(x(\cdot), t) - \phi_k \right] \cdot \right.$$

$$\left. \cdot \left[F_k(x(t)) - c_k(x(\cdot), t) + f(x(t), u, t)^T \int_{t_0^{erg}}^{t} \frac{\partial F_k(x(s))}{\partial x(s)} ds \right] \right\} \tag{18}$$

where $c_k(x(\cdot), t) = \frac{1}{t - t_0^{erg}} \int_{t_0^{erg}}^{t} F_k(x(s))ds$ denote the Fourier-parameterized spatial statistics of the state trajectory up to time t. Through simple computation, we can verify that $L(x_d(\cdot), 0, \cdot) = 0$ when $x_d(\cdot) \in Q_d$. Then, the ergodic open-loop problem improves the ergodic cost at each time step by

an amount specified by the condition (9) with $C_\mathcal{E}$ defined as

$$C_\mathcal{E} = \int_{t_{i-1}+T}^{t_i+T} L(x_i^{def}(\cdot), u_i^{def}(t), t) \, dt. \tag{19}$$

This choice of $C_\mathcal{E}$ allows us to rewrite expression (9) of the open-loop problem, as a contractive constraint used later in the Proof of Theorem 1. Contractive constraints have been widely used in the MPC literature to show closed-loop stability as an alternative to methods relying on a terminal (region) constraint [81], [89], [90]. Conditions similar to the contractive constraint used here (see (20)) also appear in terminal region methods [81], [89], [90], either in continuous or in discrete time, as an intermediate step used to prove closed-loop stability.

Next, we define stability in the ergodic sense[5].

Definition 5: Let $\mathcal{X}^\nu \subset \mathbb{R}^\nu$ be the set of states to be ergodically explored. The closed-loop solution $x_\nu(t) : \mathbb{R} \to \mathcal{X}^\nu$ resulting from an ergodic control strategy applied on (6) is ergodically stable if the difference $C(x) - \Phi(x)$ for all x with $C(x)$ defined in (1) (see Section III) converges to a zero density function $0(x)$. Using Fourier parameterization as shown in equations (2) and (3), this requirement is equivalent to $c_k(x_\nu) - \phi_k(x_\nu) \to 0$ for all k as $t \to \infty$.

The following assumptions will be necessary in proving stability.

Assumption 1: The dynamics f in (6) are continuous in u, piecewise-continuous in t, and continuously differentiable in x. Also, f is compact, and thus bounded, on any given compact sets $\mathcal{X}$ and $\mathcal{U}$. Finally, $f(\cdot, 0, 0) = 0$.

Assumption 2: There is a continuous positive definite—with respect to the set Q_d—and radially unbounded function $\mathcal{M} : Q \times \mathbb{R} \to \mathbb{R}_+$ such that $L(x(\cdot), u, t) \ge \mathcal{M}(x(\cdot), t)$ for all $u \in \mathbb{R}^m$.

Assumption 2 is necessary to show that the integral $\int_{t_0^{erg}}^{t} \mathcal{M}(x(\cdot), s)ds$ is bounded for $t \to \infty$. This result can be then used in conjunction with a well known lemma in [91]–[93] to prove convergence in the proof of the stability theorem that follows.

Theorem 1: Let assumptions 1-2 hold for all time steps i. Then, the closed-loop system resulting from the receding-horizon ergodic control strategy is ergodically stable in the sense that $c_k(x_\nu) - \phi_k(x_\nu) \to 0$ for all k as $t \to \infty$.

Proof Note that the ergodic metric (7) can be written as $J_\mathcal{E} = \mathcal{B}(t_i + T, x(\cdot))$ with $\mathcal{B}(t, x(\cdot)) := Q \sum_{k \in \mathcal{K}} \Lambda_k \left[c_k(x(\cdot), t) - \phi_k \right]^2$ with $c_k(x(\cdot), t)$ defined in Definition 4. Using this definition and converting $J_\mathcal{E}$ from Mayer to Lagrange form yields $J_\mathcal{E} = \int_{t_0^{erg}}^{t_i+T} L(x(\cdot), u, t)dt$ with $L(x(\cdot), u, t) = \frac{d}{dt}\mathcal{B}(t, x(\cdot))$ resulting in the expression in (18). Going back to Definition 4 and the ergodic open-loop problem (8) in Section IV-A, we note that condition (9) with $C_\mathcal{E}$ in (19) is a contractive constraint applied in order to generate actions that sufficiently improve the cost between time steps. To see that this is true, one can rewrite

---

[5]Note how this definition differs from the definition of asymptotic stability about an equilibrium point as we now refer to stability of a motion instead of stability of a single point.

(9) as

$$J_\mathcal{E}(x_i^*(\cdot)) - J_\mathcal{E}(x_{i-1}^*(\cdot)) \leq - \int_{t_{i-1}}^{t_i} L(x_{i-1}^*(\cdot), u_{i-1}^*(t), t)\, dt$$

$$\leq - \int_{t_{i-1}}^{t_i} \mathcal{M}(x_{i-1}^*(\cdot), s)ds \qquad (20)$$

since from (10), $u_i^{def}(t) \equiv u_{i-1}^*(t)$ in $[t_i, t_{i-1} + T]$. This contractive constraint directly proves that the integral $\int_{t_0^{erg}}^t \mathcal{M}(x(\cdot), s)ds$ is bounded for $t \to \infty$, which, according to Barbalat's lemma found, e.g., in [91]–[93], guarantees asymptotic convergence, i.e., that $x(\cdot) \to Q_d$ or equivalently that $c_k(x(\cdot)) - \phi_k \to 0$ as $t \to \infty$.

C. Multi-agent ergodic exploration

Assume we have N number of agents $\zeta = 1, ..., N$, each with its own computation, collectively exploring a terrain to track an assigned spatial distribution $\Phi(x)$. Each agent ζ performs RHEE as described in Algorithm 1. At the end of each algorithm iteration i (and thus every t_s seconds), each agent ζ communicates the Fourier-parameterized statistics of their exploration trajectories $c_{k,\zeta}^i$ up to time t_i to all the other agents. By communicating this information, the agents have knowledge of the collective coverage up to time t_i and can use this to avoid exploring areas that have already been explored by other agents. This ensures that the exploration process is coordinated so that the spatial statistics of the combined agent trajectories collectively match the distribution.

To use this information, each agent ζ updates its trajectory coefficients $c_{k,\zeta}^i$ at time t_i to include the received coefficients $c_{k,j}^{i-1}$ from the previous algorithm iteration $i-1$ so that now the collective agent coefficients $\mathbf{c}_{k,\zeta}^i$ are defined to be:

$$\mathbf{c}_{k,\zeta}^i = c_{k,\zeta}^i + \frac{1}{N-1} \cdot \sum_{j=1, j\neq\zeta}^N c_{k,j}^{i-1} \qquad (21)$$

where $c_{k,\zeta}^i$ are the coefficients of the agent ζ state trajectory at time step t_i calculated as in (17), and $c_{k,j}^{i-1}$ are the coefficients of the remaining agents state trajectories at the previous time step t_{i-1} also calculated as in (17). So now, agent ζ computes the ergodic cost (7) at t_i based on all the agents' past trajectories and Algorithm 1 is guaranteed to compute a control action that will optimally improve it. Note that expression (21) expands to:

$$\mathbf{c}_{k,\zeta}^i = \frac{1}{t_i + T - t_0^{erg}} \int_{t_0^{erg}}^{t_i+T} F_k(x_\zeta(t))dt +$$

$$\frac{1}{(N-1)(t_{i-1} + T - t_0^{erg})} \sum_{j=1, j\neq\zeta}^N \int_{t_0^{erg}}^{t_{i-1}+T} F_k(x_j(t))dt \qquad (22)$$

with $\mathbf{c}_{k,\zeta}^0 = \frac{1}{t_i + T - t_0^{erg}} \int_{t_0^{erg}}^{t_0+T} F_k(x_\zeta(t))dt$ where $x_\zeta(t)$ with $\zeta = 1, ..., N$ is the agent ζ state trajectory. Therefore expression (21) calculates the combined statistics of the current agent's
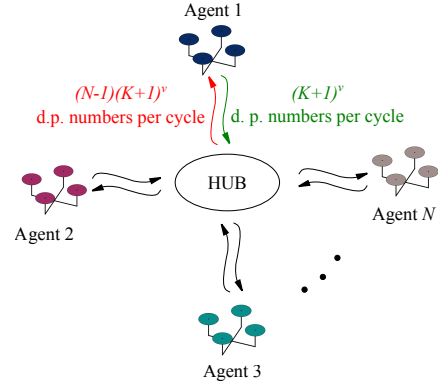

Fig. 2. Communication network for multi-agent ergodic exploration using a hub configuration. Agents are equipped with independent computational units for local control calculation but exchange information that may influence each other's subsequent actions. The HUB is simply a network component and has no computational capacity. Assuming that a double precision (d.p.) number has 64 bits and an algorithm cycle completes in t_s seconds, the transmitting bit rate of each individual communication channel should be at least $\frac{(K+1)^\nu \cdot 64}{t_s}$ bits/s and receiving bit rate equal or higher than $(N-1)\frac{(K+1)^\nu \cdot 64}{t_s}$ bits/s.

trajectory $x_\zeta(t)$, $\forall t \in [t_0^{erg}, t_i + T]$ and of the state trajectories that all the other agents have executed up to current time t_i and temporarily intend to execute from t_i (now) to $t_{i-1} + T$ based on their open-loop trajectories at the previous time step t_{i-1}.

Note that if the states of two or more agents are identical, the matching agents motion degenerates to a single agent motion and the multi-ergodic control approach fails to take advantage of all agents' control authority.

Computational complexity and communication requirements: This multi-agent ergodic control process exhibits time complexity $O(1)$ (i.e., the amount of time required for one algorithm cycle does not scale with N) because Algorithm 1 is executed by each agent in *parallel* in a distributed manner. Computational complexity also remains constant for each agent ($O(1)$), i.e., the total number of computer operations does not scale with N. However, each agent's computational unit needs to communicate with a central transmitter/receiver through a (deterministic) communication channel with receiving capacity that scales linearly in N ($O(N)$) and with constant transmitting capacity ($O(1)$). In particular, at every time step t_i, each agent needs to receive $(K + 1)^\nu$ coefficients corresponding to $c_{k,j}^{i-1}$, by each of the remaining $N - 1$ agents. In addition, each agent is responsible to transmit their own $(K+1)^\nu$ coefficients to the rest of the robot network (see Fig. 2). Thus, assuming a constant highest order of coefficients K and number of ergodic variables ν, transmitting capacity of each agent's communication channel is constant while its receiving capacity scales linearly with N. While, in this communication paradigm, we assumed a star network configuration (Fig. 2), note that a fully connected network can also be employed. In any case, the minimum amount of information needed by the team of agents for coordinated exploration is N sets of $(K+1)^\nu$ coefficients per algorithm cycle. Because of this requirement of collective data exchange between agents at each time step, multi-agent ergodic exploration can be characterized as semi-

distributed in that each agent executes RHEE (Algorithm 1) independently but shares information with the other agents after each algorithm cycle.

V. Receding-horizon ergodic target localization

This section describes the requirements for application of reactive RHEE (Algorithm 2) in target localization and tracking, starting with the formulation of the expected information density.

A. Expected Information Density

Ergodic control for localization of static or moving targets is essentially an application of reactive RHEE in Algorithm 2 with the following specifics: 1) the agent takes sensor measurements every t_m seconds while exploring the distribution in Step 4, and 2) belief of targets' state is updated online and used for computation of $\Phi(x)$ in Step 1.

We focus on the part of calculating $\Phi(x)$ (for now on referred to as Expected Information Density, EID) given the current targets belief and a known measurement model. It is important to point out that the following process for computing the EID depends only on the measurement model; the methodology for belief state representation and update can be arbitrary (e.g., Bayesian methods, Kalman filter, particle filter etc.) and does not alter the ergodic target localization process. The objective is to estimate the unknown parameters $\alpha \in \mathbb{R}^M$ describing the M coordinates of a target. We assume that a measurement $\mathbf{z} \in \mathbb{R}^\mu$ is made according to a known measurement model

$$\mathbf{z} = \Upsilon(\boldsymbol{\alpha}, \mathbf{x}) + \delta, \tag{23}$$

where $\Upsilon(\cdot)$ is a function of sensor configuration and parameters, and δ represents zero mean Gaussian noise with covariance Σ, i.e., $\delta \sim \mathcal{N}(0, \Sigma)$.

As in [14], we will use the Fisher Information Matrix (FIM) [94], [95] to calculate the EID. Often used in maximum likelihood estimation, Fisher information $\mathcal{I}(x, \boldsymbol{\alpha})$ is the amount of information a measurement provides at location x for a given estimate of $\boldsymbol{\alpha}$. It quantifies the ability of a set of random variables, in our case measurements, to estimate the unknown parameters. For estimation of parameters $\boldsymbol{\alpha} \in \mathbb{R}^M$, the Fisher information is represented as a $M \times M$ matrix. Assuming Gaussian noise, the (i, j)th FIM element is calculated as

$$\mathcal{I}_{i,j}(x, \boldsymbol{\alpha}) = \frac{\partial \Upsilon(\boldsymbol{\alpha}, x)}{\partial \alpha_i}^T \Sigma^{-1} \frac{\partial \Upsilon(\boldsymbol{\alpha}, x)}{\partial \alpha_j} \tag{24}$$

where $\Upsilon(\boldsymbol{\alpha}, \mathbf{x}) : \mathbb{R}^M \times \mathbb{R}^n \to \mathbb{R}^\mu$ is the measurement model with Gaussian noise of covariance $\Sigma \in \mathbb{R}^\mu$. Since the estimate of the target position $\boldsymbol{\alpha}$ is represented as a probability distribution function $p(\boldsymbol{\alpha})$, we take the expected value of each element of $\mathcal{I}(x, \boldsymbol{\alpha})$ with respect to the joint distribution $p(\boldsymbol{\alpha})$ to calculate the expected information matrix, $\boldsymbol{\Phi}_{i,j}(x)$. The (i, j)th element of $\boldsymbol{\Phi}_{i,j}(x)$ is then

$$\boldsymbol{\Phi}_{i,j}(x) = \int_{\alpha} \mathcal{I}_{i,j}(x, \boldsymbol{\alpha}) p(\boldsymbol{\alpha}) \, d\boldsymbol{\alpha}. \tag{25}$$

To reduce computational cost, this integration is performed numerically by discretization of the estimated parameters on a grid and a double nested summation. Note that target belief $p(\boldsymbol{\alpha})$ might incorporate estimates of multiple targets depending on the application. For that reason, this EID derivation process is independent of the number of targets and method of targets belief update.

In order to build a density map using the information matrix (25), we need a metric so that each state x is assigned a single information value. We will use the following mapping:

$$\Phi(x) = \det \boldsymbol{\Phi}(x). \tag{26}$$

The FIM determinant (D-optimality) is widely used in the literature, as it is invariant under re-parameterization and linear transformation [96]. A drawback of D-optimality is that it might result in local minima and maxima in the objective function, which makes optimization difficult when maximizing information. In our case though, local maxima do not pose an issue as our purpose is to approximate the expected information density using ergodic trajectories instead of maximizing it.

B. Remarks on Localization with Limited Sensor Range

The efficiency of planning sensor trajectories by maximizing information metrics like the Fisher Information Matrix in (24) is highly dependent on the true target location [96]: if the true target location is known, the optimized trajectories are guaranteed to acquire the most useful measurements for estimation; if not, the estimation and optimization problems must be solved simultaneously and there is no guarantee that useful measurements will be acquired especially when the sensor exhibits limited range.

A limited sensor range serves as an occlusion during localization, in that large regions are naturally occluded while taking measurements. Because of this, how we plan the motion of the agent according to the current target estimate is critical; if, at one point, the current target belief largely deviates from the true target position, the sensor might completely miss the actual target (out of range), never acquiring new measurements in order to update the target's estimate. This would be a possible outcome if we controlled the agent to move towards maximum information (IM). In this section, we explain how receding-horizon ergodic target localization (Algorithm 2) with limited sensor range can overcome this drawback under a single assumption.

Assumption 3: Let $r \in \mathbb{R}^+$ be the radius defining sensor range so that a sensor positioned at $\boldsymbol{x}_s \in \mathcal{X}_\nu$ can only take measurements of targets whose true target location $\boldsymbol{\alpha}_{true} \in \mathbb{R}^\nu$ satisfies $\|\boldsymbol{x}_s - \boldsymbol{\alpha}_{true}\|_\nu < r$. An occlusion O is defined as the region where no sensing occurs i.e., $O = \{x_s \in \mathcal{X}_\nu : \|\boldsymbol{x}_s - \boldsymbol{\alpha}_{true}\|_\nu > r\}$. At all times $t_{curr} < \infty$ in Algorithm 2, there is $x_q \in \mathcal{X}_\nu$ that simultaneously satisfies $\|\boldsymbol{x}_q - \boldsymbol{\alpha}_{true}\|_\nu < r$ and $\Phi_{curr}(x_q) > 0$, where $\Phi_{curr}(x) \forall x \in \mathcal{X}_\nu$ is the expected information density computed as in (26) at time t_{curr}.

Proposition 2: Let Assumption 3 hold. Also, let $x_\nu(\cdot) : [t_{curr}, \infty) \to \mathcal{X}_\nu$ denote the exploration trajectory of an agent performing ergodic target localization (Algorithms 1 and 2)

with expected information density $\Phi_{curr}(x) \forall x \in \mathcal{X}_\nu$, equipped with a sensor of range $r \in \mathbb{R}$. Then, there will be time $t_s \in [t_{curr}, \infty)$ where the agent's state satisfies $x_\nu(t_s) \in \mathcal{X}_\nu \backslash O$, so that new measurements are acquired and $\Phi_{curr}(x)$ is updated.

Proof Due to Assumption 3, at time t_{curr} there is $x_q \in \mathcal{X}_\nu \backslash O$ that satisfies $\Phi_{curr}(x_q) > 0$. According to Theorem 2 and Definition 5, it is $C(x) - \Phi_{curr}(x) \rightarrow 0$ for all $x \in \mathcal{X}_\nu$ as $t \rightarrow \infty$. Therefore, at some time $t \in [t_{curr}, \infty)$, we know that $C(x_q) = \Phi_{curr}(x_q) > 0$ that is equivalent to $\frac{1}{t - t_{curr}} \int_{t_{curr}}^{t} \delta[x_q - x_\nu(\tau)] d\tau > 0$ from Eq. (1). This leads to the conclusion that $x_q \in x_\nu(\cdot)$ which directly proves the proposition.

Assumption 4—stating that information density is always non-zero in an arbitrarily small region around the true target—can be satisfied in various ways. For example, we can adjust the parameters of the estimation filter to achieve a sufficiently low convergence rate. Alternatively, in cases of high noise and variability, we can artificially introduce nonzero information values across the terrain so as to promote exploration as in the simulation and experimental examples.

VI. SIMULATION RESULTS

A. Ergodic Exploration and Coverage

1) Motivating example - Uniform area coverage with occlusions: In this first example, we control an agent to explore an occluded environment in order to achieve a uniform probability of detection across a square terrain, using Algorithm 1. The shaded regions (occlusions) O comprise a circle and a rectangle in Fig. 3 and they exhibit zero probability of detection i.e., $\Phi(x) = 0 \forall x \in O$. Such situations can arise in vision-based UAV exploration with occlusions corresponding to shaded areas that limit visibility, or in surveillance by mobile sensors
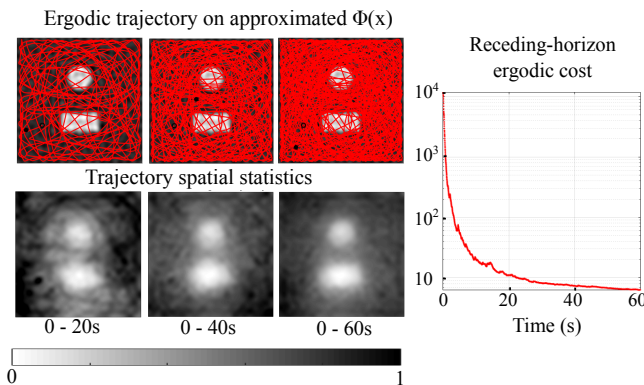

Fig. 3. Ergodic area coverage in an occluded environment (Algorithm 1 with time horizon $T = 0.1s$ and sampling time $t_s = 0.02s$). White regions in $\Phi(x)$ (top row) indicate low to no probability of detection (occlusions), for example due to sensor failure or physical entities obscuring visibility. Note that occlusions are not obstacles that should be completely avoided. Bottom row shows the spatial statistics $\Phi_x^i(x)$ of the followed trajectory from $t = 0$ to $t = t_i$ calculated as $\Phi_x^i(x) = \sum_{k \in \mathcal{K}} \{\Lambda_k c_k^i F_k(x)\}$ with $\nu = 2$ and $K = 20$. By the end of the simulation at $t = 60$, the trajectory spatial statistics $\Phi_x^{60}(x)$ closely match the initial terrain spatial distribution $\Phi(x)$, accomplishing the objective of ergodicity as expected. The ergodic cost (7) is shown to decrease on logarithmic scale over time. Small cost fluctuations result from numerical errors.

where the shaded regions can be thought of as areas where no sensor measurements can be made due to foliage. It is assumed that the agent has second-order dynamics with $n = 4$ states $x = [x_1, x_2, x_3, x_4]$, $m = 2$ inputs $u = [u_1, u_2]$, and $f(x, u) = [x_2, u_1, x_4, u_2]$ in (6). Forcing saturation levels are set as $u_{min} = -50$ and $u_{max} = 50$.

Snapshots of the agent exploration trajectory is shown in Fig. 3. As time progresses from $t = 0$ to $t = 60$ the spacing between the trajectory lines is decreasing, meaning that the agent successfully and completely covers the square terrain by the end of the simulation. This is also reflected in the spatial statistics of the performed trajectory that eventually closely match the desired probability of detection. A similar example was used by Mathew and Mezić in [73] for evaluation of their ergodic control method that was specific to double integrator systems. Our results serve as proof of concept, showing that Algorithm 1—although designed to control complex nonlinear systems—can still handle simple systems efficiently and achieve full area coverage, as expected.

2) Multi-agent aerial exploration: The previous example showed how RHEE can perform area coverage using the simple double-integrator dynamic model. Here and for the rest of this section, we will utilize a 12-dimensional quadrotor model to demonstrate the algorithm's efficiency in planning trajectories for agents governed by higher-dimensional nonlinear dynamics. The search domain is two-dimensional with $\nu = 2$. The quadrotor model [97]–[99] has 12 states ($n = 12$ in system (6)), consisting of the position $[x_q, y_q, z_q]$ and velocity $[\dot{x}_q, \dot{y}_q, \dot{z}_q]$ of its center of mass in the inertial frame, and the roll, pitch, yaw angles $[\phi_q, \theta_q, \psi_q]$ and corresponding angular velocities $[\dot{\phi}_q, \dot{\theta}_q, \dot{\psi}_q]$ in the body frame. Each of the 4 motors produces the force u_i, $i = 1, ..., 4$ ($m = 4$ in (6)), which is proportional to the square of the angular speed, that is, $u_i = k\omega^2$. Saturation levels are set as $u_{min} = 0$ and $u_{max} = 12$ in (6). Nominal control u_i^{nom} from equation (10) in Algorithm 1 is a PD (proportional-derivative) controller that regulates the agent's height to maintain a constant value.

We use five aerial vehicles to collectively explore a terrain based on a constant distribution of information. The resulting agent trajectories and corresponding spatial statistics are shown in Fig. 4. It is important to notice here that each agent is not separately assigned to explore a single distribution peak (as a heuristic approach would entail) but rather all agents are provided with the same spatial distribution as a whole and their motion is planned simultaneously in real time to achieve best exploration on the areas with highest probability of detection.

This simulation example was coded in C++ and executed at a Linux-based laptop with an Intel Core i7 chipset. Assuming that each quadrotor executes Algorithm 1 in parallel, the execution time of the $120s$ simulation is approximately $\sim 70s$ per quadrotor, running about two times faster than real time.

B. Ergodic Coverage and Target Localization

In the following examples, we will use the 12-dimensional nonlinear quadrotor model from the previous section to perform motion planning for vision-based static and moving target localization with bearing-only sensing through a gimbaled

Ergodic trajectories on approximated $\Phi(x)$

Ergodicity

Trajectories spatial statistics
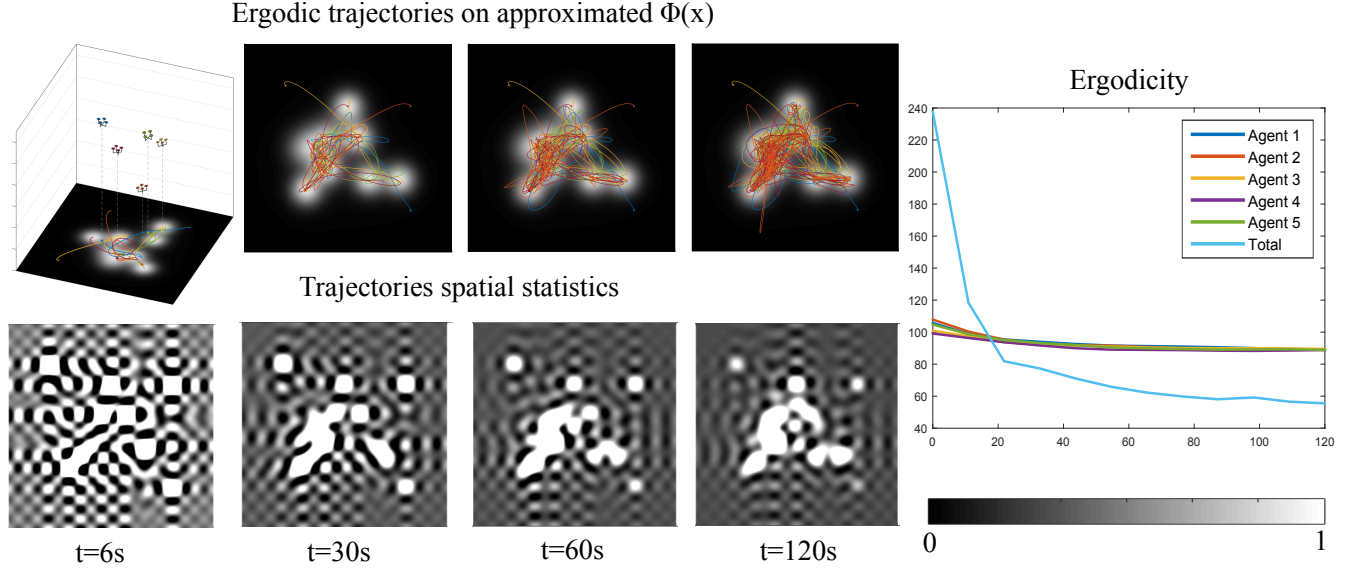
t=6s   t=30s   t=60s   t=120s

Fig. 4. Multi-agent UAV exploration (each agent executes Algorithm 1 with trajectory coefficients calculated as in (21) with $N = 2$, time horizon $T = 1.3s$ and sampling time $t_s = 0.1s$). Five quadrotor models collectively explore a terrain to track a spatial distribution $\Phi(x)$ (top row). Highest order of coefficients is $K = 12$. Note that agents naturally avoid exploring the same region simultaneously and only return back to already visited areas when sufficient time has passed. Bottom row shows the spatial statistics of the combined agent trajectories calculated as described in Fig. 3 caption. As expected, by the end of simulation, the collective spatial statistics match closely the initial spatial distribution. Plot on the right shows the ergodicity measure of trajectories as they evolve in time. Ergodicity of each agent's trajectory at time t is calculated as $\sum_k \{ \Lambda_k [\frac{1}{t-t_0^{erg}} \int_{t_0^{erg}}^t F_k(x_\zeta(s))ds - \phi_k]^2 \}$ for the ergodic trajectories $x_\zeta(t)$ with $\zeta = 1, ..., N$. Total ergodicity of the collective trajectories is calculated as $\sum_k \{ \Lambda_k [\frac{1}{t-t_0^{erg}} \int_{t_0^{erg}}^t \sum_{j=1}^N F_k(x_j(s))ds - \phi_k]^2 \}$. A video representation of this exploration process is available in the supporting multimedia files.

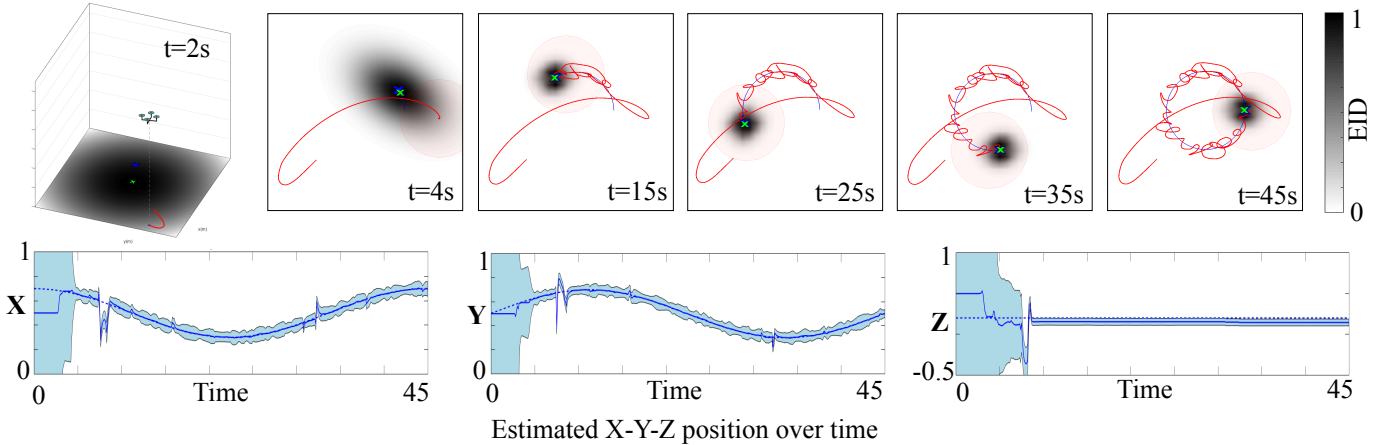Estimated X-Y-Z position over time

Fig. 5. Bearing-only localization of a moving target. Top: Top-view snapshots of the UAV trajectory (red curve) where the true target position (blue X-mark) and path (blue curve), and the estimated target position (green X-mark) are also illustrated. The quadrotor can acquire vision-based measurements with a limited range of view that is illustrated as a light red circle around the current UAV position. No prior behavior model of the target motion is available for estimation using EKF. The limited sensor range serves as an occlusion as it naturally occludes large regions while taking measurements. The highest order of coefficients is $K = 10$. The quadrotor explores the areas with highest information to acquire useful measurements. Although the geometry of the paths is not predefined, the resulting trajectories follow a cyclic, swirling pattern around the true target position, as one would naturally expect — like in standoff tracking solutions for example [26]. Bottom: The target estimate (solid blue curve) is compared to the real target position (dashed blue curve) along with an illustration of the belief covariance (light blue area around estimated position) over time. The target belief converges to a normal spatial distribution with the mean at the true target position and low covariance. A video representation of this exploration process is available in the supporting multimedia files.

camera that always faces in the direction of gravity. Representing the estimate of target's position as a Gaussian probability distribution function, we use an Extended Kalman Filter (EKF) [100], [101] to update the targets' position belief based on the sensor measurements. We use EKF because it is fast, easy to implement and regularly used in real-world applications but any other estimator (e.g., the Bayesian approaches in [102]) can be used instead with no change to the process of Algorithm 2. Note for example that reliable bearings-only estimation with EKF cannot be guaranteed as previous results indicate [103], so it might be desirable to use a more specialized estimator.
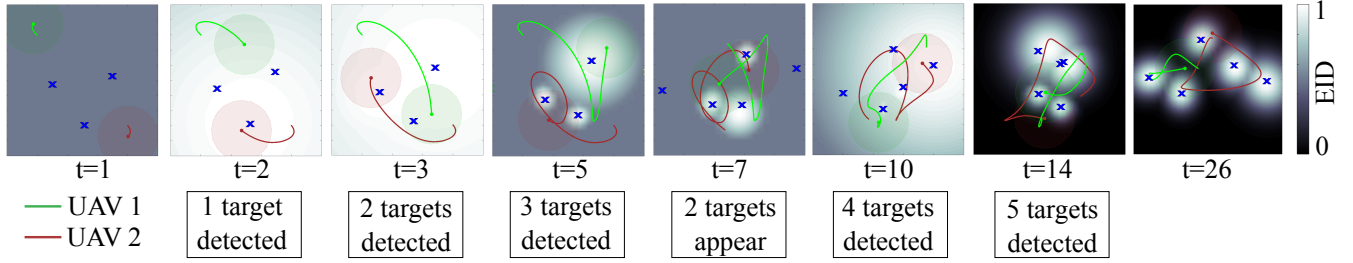
Fig. 6. Multi-agent simultaneous exploration and targets localization. The problem of exploration vs exploitation is addressed by controlling two agents to localize detected targets while exploring for new undetected targets. The algorithm scales to multiple target localization without any modification, as it tracks a universal non-parametric information distribution instead of each target independently. For cleaner representation, only the UAV trajectories of the past $5s$ are shown in each snapshot. Highest order of coefficients is $K = 10$. Mean and standard deviation of targets belief (not shown here) fluctuate in a pattern similar to the experimental results in Fig. 10. Light green and red circles around the current UAV positions indicate the camera range of view. Notice that the EID value is set at a middle level (gray color) in areas where no high information measurements can be taken from the already detected targets. This serves to promote exploration for more targets.

It is assumed that the camera uses image processing techniques (e.g., [104]) to take bearing-only measurements, measuring the azimuth and elevation angles from current UAV position $[x_q, y_q, z_q]$ to the detected target's position $[x_\tau, y_\tau, z_\tau]$. The corresponding measurement model is $\mathbf{z}$ in (23) with

$$\Upsilon([x_\tau, y_\tau, z_\tau], [x_q, y_q, z_q]) = \begin{bmatrix} \tan^{-1}\left(\frac{x_q - x_\tau}{y_q - y_\tau}\right) \\ \tan^{-1}\left(\frac{z_q - z_\tau}{\sqrt{(x_q - x_\tau)^2 + (y_q - y_\tau)^2}}\right) \end{bmatrix}. \quad (27)$$

The measurement noise covariance is $\Sigma = diag\{[0.1, 0.1]\}$ in radians. As in the previous examples, a PD controller serves as nominal control, regulating UAV height z_q. The target transition model for EKF is expressed as $\alpha_i = \mathcal{F}(\alpha_{i-1}) + \epsilon$ with ϵ representing zero mean Gaussian noise with covariance C, i.e., $\epsilon \sim \mathcal{N}(0, C)$. We assume that no prior behavior model of the target motion is available and thus the transition model is $\mathcal{F}(\alpha_{i-1}) \equiv \alpha_{i-1}$. Importantly, the camera sensor has limited range of view, completely disregarding targets that are outside of a circle centered at the UAV position with constant radius (as depicted in Fig. 5).

1) Bearing-only localization of a moving target with limited sensor range: Here, we demonstrate an example where a quadrotor is ergodically controlled to localize a moving target, with frequency of measurements $f_m = 20$Hz and frequency of EID update at $f_\phi = 10$Hz. The 3D target position $[x_\tau, y_\tau, z_\tau]$ is localized so that $M = 3$. We assume that no prior behavior model of the target motion is available and thus the transition model is $F(\alpha_{k-1}) \equiv \alpha_{k-1}$ with covariance $C = diag\{[0.001, 0.001, 0.001]\}$ (i.e modeled as a diffusion process). Top-view snapshots of the UAV motion are shown in Fig. 5. The agent detects the target without prior knowledge of its position, whereafter it closely tracks the target by applying Algorithm 2 to adaptively explore a varying expected information density $\Phi(x)$. Although the geometry of the paths is not predefined, the resulting trajectories follow a cyclic, swirling pattern around the true target position, as one might expect.

This simulation example was coded in C++ and executed at a Linux-based laptop with an Intel Core i7 chipset. The execution time of the $45s$ simulation is approximately $\sim 25s$. This result is representative of the algorithm's computational cost and execution time, because it involves a high-dimensional, nonlinear system. Localization is slower than pure exploration, mainly because it requires calculation of the expected information density every t_ϕ seconds using the expressions (24), (25) and (26).

2) Multi-agent simultaneous terrain exploration and target localization: This simulation example is designed to demonstrate search for undetected targets (exploration) and localization of detected moving targets simultaneously, using two agents. A random number of targets must be detected (exploration) and tracked (target localization) by two UAVs. Note that here we do not address the issue of cooperative sensing filters [105] for multiple sensor platforms: instead, we use a centralized Extended Kalman Filter for simplicity but any filter that provides an estimate of the target's state can be employed instead.

Top-view snapshots of the multi-agent exploration trajectories are given in Fig. 6. At $t = 0$ when 3 targets are present in the terrain but none of them have been detected by the agents, the EID is a uniform distribution across the workspace. Information density is set at $\Phi(x) = 0.5$ for all x (gray coloring). By $t = 5$ all present (three) targets have been detected and the EID map is computed based on Fisher Information using expressions (24), (25) and (26). Information density is still set at a middle level (instead of zero) in areas where information of target measurements is zero. This serves to promote exploration in addition to localization. In this special case, the terrain spatial distribution $\Phi(x)$ is defined to encode both probability of detection (for the undetected targets) and expected information density (for the detected targets). At $t = 7$ two more targets appear and by $t = 14$ five targets have been detected. Here, we assume that no more than five targets are to be detected and thus, after the fifth target detection, the spatial distribution only encodes expected information density (note that $\Phi(x) = 0$ for all x where information from measurements is zero).

This simulation example was coded in C++ and executed at a Linux-based laptop with an Intel Core i7 chipset. Assuming that each quadrotor executes Algorithm 1 in parallel, the
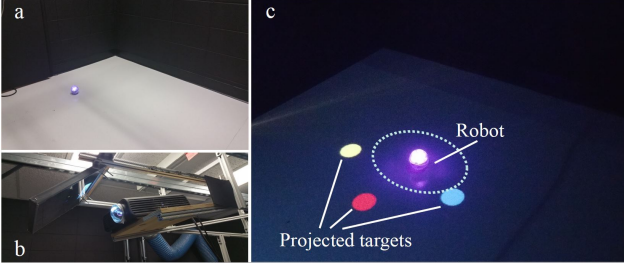
Fig. 7. (a) The `sphero` SPRK robot is shown in the experimental setup. The internal mechanism shifts the center of mass by rolling and rotating within the spherical enclosure. RGB LEDs on the top of the `sphero` SPRK are utilized to track the odometry of the robot through a webcam using OpenCV for motion capture. The Robot Operating System (ROS, available online [106]) is used to transmit and collect data at 20 Hz. A projection (b) is used to project the targets onto the experimental floor shown in (c).
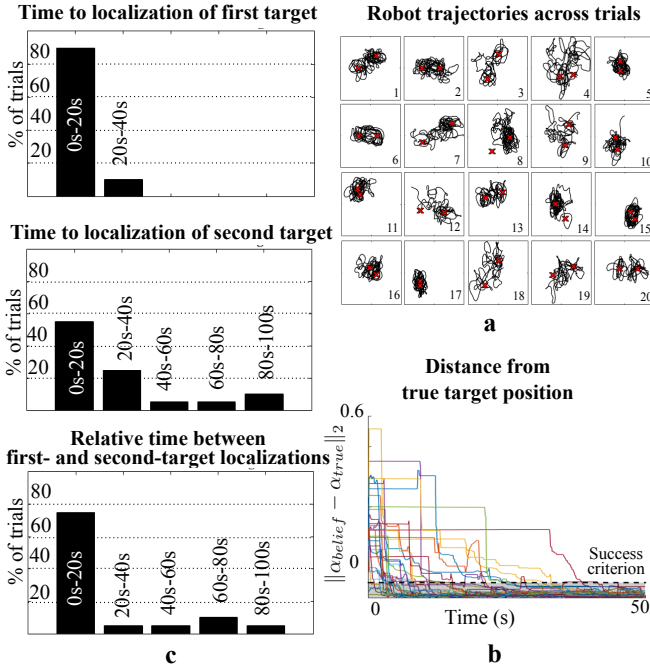

Fig. 9. The `sphero` SPRK robot robustly localizes increasing number of moving targets with bearing-only measurements. The targets' belief—represented as a black-and-white spatial distribution—remains close to the actual targets' location for all 280 seconds. The robot trajectories of a 40-second time window are also shown in red.

Fig. 8. Twenty trials of localizing 2 random targets using the `sphero` SPRK robot at a 1m×1m terrain with simulated limited sensor range of 0.2m. a) Top-view snapshots of the robot trajectories (black) across trials. The algorithm robustly localizes random pairs of static targets (shown in red) by performing cyclic trajectories around the targets (as required for useful bearing-only measurements), without path specification—the behavior results naturally from the objective of improving ergodicity with respect to the expected information density. b) The distance of the mean target estimate from true target position over time across all trials that were complete by the first 50 seconds. Distance remains constant for as long as the target is outside of the sensor range or it has not be detected yet. c) Bar graphs showing time to localization of first target (top), of second target (center) and relative time that the second target was localized after the first target (bottom), across trials. The localization of a target is defined to be successful when the ℓ^2-norm of the difference between the target's position belief and the real target position falls below 0.05, i.e., $\|\alpha_{belief} - \alpha_{true}\|_2 < 0.05$. In 100% of the trials, the first target is localized within 40 seconds. In 80% of the trials, both targets are localized by the first 40 seconds. In 75% of the trials, the waiting time between localizing the first and second target is less than 20 seconds. Even when target detection is delayed or the EKF fails to converge in a few iterations, the robot is successful in localizing all the targets by 100 seconds.

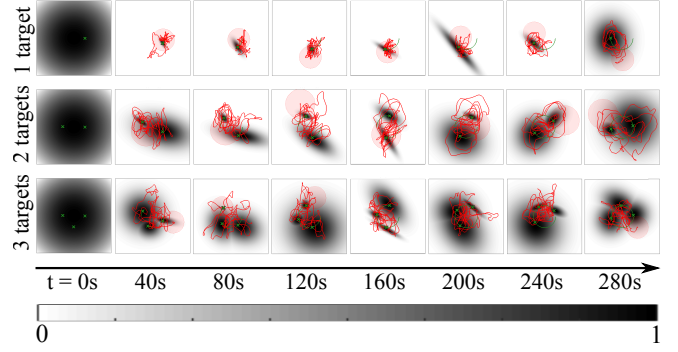execution time is approximately $\sim 30s$ per quadrotor.

VII. Experimental Results

We perform two bearing-only target localization experiments using a `sphero` SPRK robot [17] in order to verify real-time execution of Algorithm 1 and showcase the robustness of the algorithm in bearing-only localization. In addition to the robot, the experimental setup involves an overhead projector and a camera, and is further explained in Fig. 7. The overhead camera is used to acquire sensor measurements of current robot and target positions that are subsequently transformed to bearing-only measurements as in (27). We additionally simulate a limited sensor range as a circle of 0.2 m radius around the current robot position. As in the quadrotor simulation examples, we use an Extended Kalman Filter for bearing-only estimation. In all the following experiments, the ergodic controller runs at approaximately 10Hz frequency, i.e., $t_s = 0.1s$ in Algorithm 1.

A. Experiment 1

In this Monte Carlo experiment, we perform twenty trials of localizing two static targets randomly positioned in the terrain. For each trial, we consider the localization of a target successful when the ℓ^2-norm of the difference between the target's position belief and the real target position falls below 0.05, i.e., $\|\alpha_{belief} - \alpha_{true}\|_2 < 0.05$. To promote variability, initial mean estimates of target positions are also randomly selected for each trial. Initial distribution $\Phi(x)$ is uniform inside the terrain boundaries.

The robot simultaneously explores the terrain for undetected targets and localizes detected targets. As in the simulation example of Section VI-B2, we achieve simultaneous exploration and localization by setting the probability of detection (i.e., distribution $\Phi(x)$) across the terrain at a nonzero value. For most trials, targets are successfully localized in less than 60 seconds. We see that even in the few cases when target detection is delayed due to limited sensor range or when the EKF fails to converge (as expected for bearing-only models [103]) (see trials with time to second-target localization higher than 60s in Fig. 8c), the robot manages to eventually localize both targets by fully exploring the EID instead of moving towards the EID maximum as in information maximization
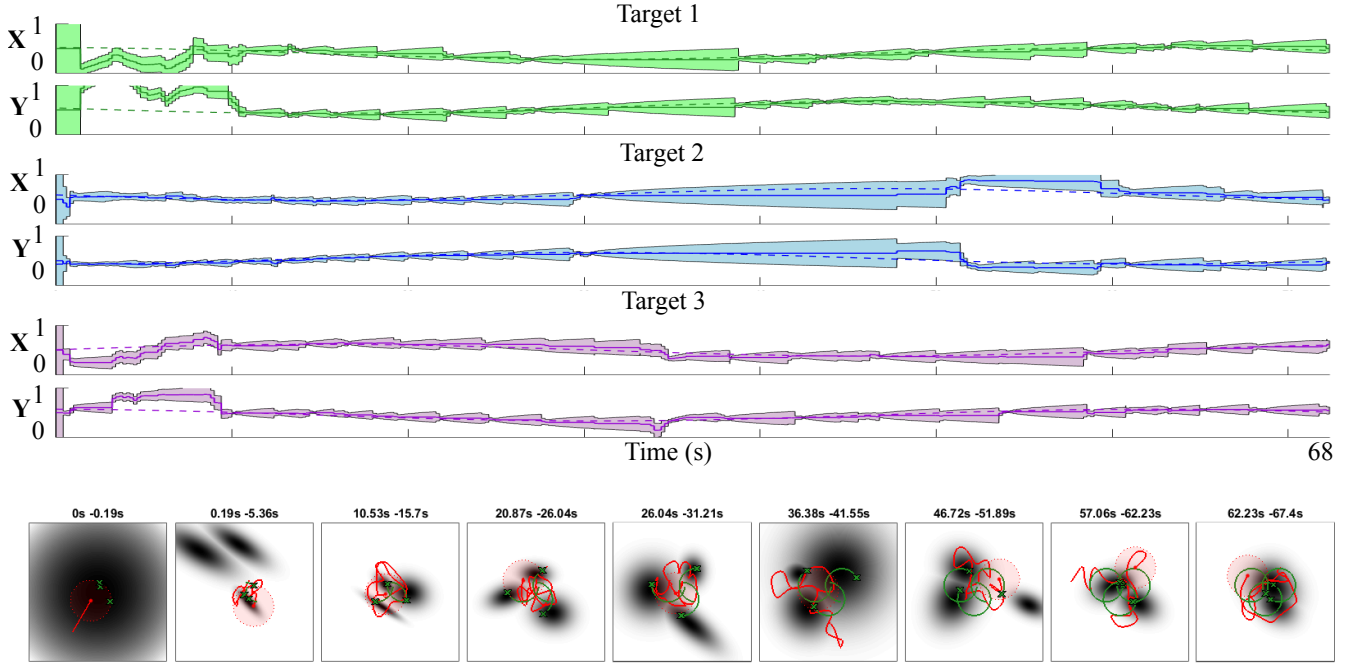
Fig. 10. Localization of 3 moving targets using the `sphero` SPRK robot. The target estimates (solid curves) are compared to the real target locations (dashed curves) along with an illustration of the belief covariance (shaded area around estimated position) over time. Because the targets are constantly moving and the sensor range is limited, the standard deviation of the targets belief fluctuates as time progresses. The agent localizes each target alternately; once variance on one target estimate is sufficiently low, the agent moves to the next target. Importantly, this behavior is completely autonomous, resulting naturally from the objective of improving ergodicity. Note that we can only decompose the targets belief into separate target estimates because of our choice to use EKF where each target's belief is modeled as a normal distribution. This would not be necessarily true, had we used a different estimation filter (e.g., particle filter). Bottom row shows top-view snapshots of the robot and target's motion. A video of this experiment is available in the supporting multimedia files.

techniques. This result validates Proposition 2 that provides convergence guarantees even with poor target estimates and limited sensor range.

B. Experiment 2

With this experiment, we aim to demonstrate the robustness of the algorithm in localizing increasing number of moving targets. The resulting robot trajectories for localizing 1, 2 and 3 targets are shown in Fig. 9, while Fig. 10 shows the results for localizing 3 targets moving at a different pattern. As in the simulation examples, the motion of each target is modeled as a diffusion process. Note that because the targets are constantly moving and the sensor range is limited, the standard deviation of the targets belief fluctuates as time progresses (see Fig. 10). The agent localizes each target alternately; once variance on one target estimate is sufficiently low, the agent moves to the next target. Importantly, this behavior is completely autonomous, resulting naturally from the objective of improving ergodicity.

VIII. Discussion

A. Number of Fourier coefficients

In this subsection, we briefly discuss the effect of K, the highest order of coefficients ϕ_k in (2) and c_k in (3), on the algorithm's performance. First, note that, because both the trajectory statistics and the desired search density are parameterized by the same number of Fourier coefficients K,

all choices of K lead to stability as defined in Definition 5. However, since convergence only concerns the trajectory statistics, i.e., $c_k(x_v) - \phi_k(x_v) \to 0$, and not the trajectory itself, different choices of K can affect the computed trajectories.

Figure 11 shows the spatial statistics of a semi-circular trajectory with constant velocity, represented with an increasing number of Fourier coefficients. The representation of the agent's trajectory becomes more diffuse, as K decreases. If the search density has fine-scale details, small K might mean that the agent will disregard the details despite meeting the ergodicity requirement. In addition, as expected, there is a diminishing returns property: the rate of change in the algorithm output decreases as K is increased (because the coefficients in the ergodic metric become small as K becomes large). We can see this in Fig. 11 where the trajectory spatial statistics for $K = 30$ have nearly converged to the original agent trajectory shape.
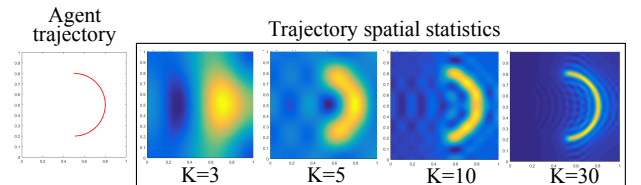

Fig. 11. The Fourier-parameterized spatial statistics of the semi-circular trajectory shown on the left, calculated for different values of the highest order of coefficients K. Yellow indicates high statistical coverage and blue is low to no coverage.

To sum up, our choice of K depends on how refined the given search density is. In the simulation and experimental results presented here, we found that a minimum $K = 5$ is sufficient in completing the assigned tasks.

B. Convergence Rate

Note from the stability analysis in Section IV-B that Algorithm 1 is formulated so that an upper bound on the convergence rate is satisfied in equation (20), determined by $\mathcal{M}(x(\cdot), t)$. Although $\mathcal{M}$ is state-dependent, we may assume an additional constraint $\mathcal{D}(t)$ in Assumption 2 so that $L(x(\cdot), u, t) \geq \mathcal{M}(x(\cdot), t) \geq \mathcal{D}(t)$. This imposes a time-dependent convergence rate of the form $J_{\mathcal{E}}(x_i^*(\cdot)) - J_{\mathcal{E}}(x_{i-1}^*(\cdot)) \leq -\int_{t_{i-1}}^{t_i} \mathcal{D}(t)\, dt$ in (20). An upper-bound on convergence rate is then the highest rate, determined by $\mathcal{D}(t)$, for which there always exists a single control action of finite duration that satisfies the constraint requirement, so that open-loop problem (8) attains a solution at each time step.

One way to satisfy this requirement is by manipulating the sampling time t_S. Our theoretic results in [93] show that—at least, for Bolza form control objectives—as long as the contractive constraint depends on t_s, there exists t_s as $t_s \to 0$ that guarantees that there is always a finite-duration single control action to satisfy the constraint requirement.

IX. Conclusions

In this paper, we exploited the advantages of hybrid systems theory to formulate a receding-horizon ergodic control algorithm that can perform real-time coverage and target localization, adaptively using sensor feedback to update the expected information density. In target localization, this ergodic motion planning strategy controls the robots to track a non-parameterized information distribution across the terrain instead of individual targets independently, thus being completely decoupled from the estimation process and the number of targets. We demonstrated—in simulation with a 12-dimensional UAV model and in experiment using the sphero SPRK robot—that ergodically controlled robotic agents can reliably track moving targets in real time based on bearing-only measurements even when the number of targets is not known a priori and the targets' motion is only modeled as a diffusion process. Finally, the simulation and experiment examples served to highlight the importance of and to verify stability of the ergodic controls with respect to the expected information density, as proved in our theoretical results.

Appendix A
Proof

The proof of Proposition 1 is as follows. To make explicit the dependence on action A, we write inputs $u : \mathbb{R} \times \mathbb{R}^+ \times \mathbb{R} \times U \to U$ of the form of $u_i^*(t)$ in (8) as

$$u(t; \lambda_A, \tau_A, u_A) = \begin{cases} u_A & \tau_A \leq t \leq \tau_A + \lambda_A \\ u_i^{def} & \text{else.} \end{cases}$$

When $\lambda_A = 0$, it is $u(t; 0, \cdot, \cdot) \equiv u_i^{def}$, i.e., no action is applied. Accordingly, we define $\bar{J}_{\mathcal{E}}(\lambda_A, \tau_A, u_A) :=$

$J_{\mathcal{E}}(x(t; t_0, x_0, u(t; \lambda_A, \tau_A, u_A)))$ so that the performance cost depends directly on the application parameters of an action A. Assuming $t_0^{erg} = t_0$ and defining $\beta := \frac{1}{t_i + T - t_0} \int_{t_0}^{t_i + T} F_k(x(t))dt - \phi_k$, it is

$$\frac{\partial J_{\mathcal{E}}}{\partial \lambda} = \frac{\partial J_{\mathcal{E}}}{\partial \beta} \frac{\partial \beta}{\partial \lambda} \tag{28}$$

where

$$\frac{\partial J_{\mathcal{E}}}{\partial \beta} = 2Q \sum_{k \in \mathcal{K}} \Lambda_k \Big[\underbrace{\frac{1}{t_i + T - t_0} \int_{t_0}^{t_i + T} F_k(x(\sigma))d\sigma - \phi_k}_{c_k^i} \Big] \tag{29}$$

and

$$\frac{\partial \beta}{\partial \lambda} = \frac{1}{t_i + T - t_0} \int_{\tau}^{t_i + T} \frac{\partial F_k(x(t))}{\partial x(t)} \frac{\partial x(t)}{\partial \lambda} dt \tag{30}$$

where the integral boundary changed from t_0 to τ because the derivative of $x(t)$ with respect to λ is zero when $t < \tau$. Then, expression (28) can be rearranged, pulling $\frac{\partial J_{\mathcal{E}}}{\partial \beta}$ into the integral over t, and switching the order of the integral and summation, to the following:

$$\frac{\partial J_{\mathcal{E}}}{\partial \lambda} = \int_{\tau}^{t_i + T} \underbrace{\frac{2Q}{t_i + T - t_0} \sum_{k \in \mathcal{K}} \Big\{ \Lambda_k [c_k^i - \phi_k] \frac{\partial F_k(x(t))}{\partial x(t)} \Big\}}_{\ell(t, x)} \frac{\partial x(t)}{\partial \lambda} dt \tag{31}$$

with[6]

$$\frac{\partial x(t)}{\partial \lambda} = \Phi(t, \tau) \Big[f(\tau, x(\tau), u_A) - f(\tau, x(\tau), u_i^{def}(\tau)) \Big] \tag{32}$$

where $\Phi(t, \tau)$ is the state transition matrix of the linearized system dynamics (6) with $A = D_x f$. Therefore,

$$\frac{\partial J_{\mathcal{E}}}{\partial \lambda} = \int_{\tau}^{t_i + T} \ell(t, x) \cdot \Phi(t, \tau) dt \cdot \Big[f(\tau, x(\tau), u_A) - f(\tau, x(\tau), u_i^{def}(\tau)) \Big]. \tag{33}$$

Finally, notice that $\int_{\tau}^{t_i + T} \ell(t, x) \cdot \Phi(t, \tau) dt$ is the convolution equation for the system

$$\dot{\rho}_{\mathcal{E}} = -\ell(t, x)^T - D_x f\big(t, x, u_i^{def}\big)^T \rho_{\mathcal{E}} \tag{34}$$

$$\text{subject to } \rho_{\mathcal{E}}(t_i + T) = \mathbf{0}$$

where ℓ is defined in (31). Therefore, we end up with the expression for the mode insertion gradient of the ergodic cost at time τ:

$$\frac{\partial J_{\mathcal{E}}}{\partial \lambda}\Big|_{\tau} = \rho_{\mathcal{E}}(\tau) \Big[f(\tau, x(\tau), u_A) - f(\tau, x(\tau), u_i^{def}(\tau)) \Big].s \tag{35}$$

References

[1] G. E. Jan, C. Luo, L. P. Hung, and S. T. Shih, "A computationally efficient complete area coverage algorithm for intelligent mobile robot navigation," in *2014 International Joint Conference on Neural Networks*, July 2014, pp. 961–966.

[6]Expression (32) is a direct result of applying the fundamental theorem of calculus on the system equations (6).

[2] Y. Stergiopoulos and A. Tzes, "Spatially distributed area coverage optimisation in mobile robotic networks with arbitrary convex anisotropic patterns," *Automatica*, vol. 49, no. 1, pp. 232–237, 2013.

[3] D. E. Soltero, M. Schwager, and D. Rus, "Decentralized path planning for coverage tasks using gradient descent adaptive control," *The International Journal of Robotics Research*, vol. 33, no. 3, pp. 401–425, 2014.

[4] R. J. Meuth, E. W. Saad, D. C. Wunsch, and J. Vian, "Adaptive task allocation for search area coverage," in *IEEE International Conference on Technologies for Practical Robot Applications*, 2009, pp. 67–74.

[5] M. Schwager, D. Rus, and J.-J. Slotine, "Decentralized, adaptive coverage control for networked robots," *The International Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.

[6] J.-M. Passerieux and D. Van Cappel, "Optimal observer maneuver for bearings-only tracking," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 34, no. 3, pp. 777–788, 1998.

[7] P. Skoglar, U. Orguner, and F. Gustafsson, "On information measures based on particle mixture for optimal bearings-only tracking," in *IEEE Aerospace conference*, 2009, pp. 1–14.

[8] A. N. Bishop and P. N. Pathirana, "Optimal trajectories for homing navigation with bearing measurements," in *Proceedings of the 2008 International Federation of Automatic Control Congress*.

[9] X. Liao and L. Carin, "Application of the theory of optimal experiments to adaptive electromagnetic-induction sensing of buried targets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 961–972, 2004.

[10] F. Bourgault, A. Göktoğan, T. Furukawa, and H. F. Durrant-Whyte, "Coordinated search for a lost target in a Bayesian world," *Advanced Robotics*, vol. 18, no. 10, pp. 979–1000, 2004.

[11] S. S. Ponda, R. M. Kolacinski, and E. Frazzoli, "Trajectory optimization for target localization using small unmanned aerial vehicles," in *AIAA Guidance, Navigation, and Control Conference*, 2009, pp. 10–13.

[12] Y. Oshman and P. Davidson, "Optimization of observer trajectories for bearings-only target localization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 35, no. 3, pp. 892–902, 1999.

[13] B. Grocholsky, A. Makarenko, and H. Durrant-Whyte, "Information-theoretic coordinated control of multiple sensor platforms," in *IEEE International Conference on Robotics and Automation*, vol. 1, 2003, pp. 1521–1526.

[14] L. M. Miller, Y. Silverman, M. A. MacIver, and T. D. Murphey, "Ergodic exploration of distributed information," *IEEE Transactions on Robotics*, vol. 32, no. 1, pp. 36–52, 2015.

[15] R. O'Flaherty and M. Egerstedt, "Optimal exploration in unknown environments," in *IEEE International Conference on Intelligent Robots and Systems*, 2015, pp. 5796–5801.

[16] L. M. Miller and T. D. Murphey, "Optimal planning for target localization and coverage using range sensing," in *IEEE International Conference on Automation Science and Engineering*, 2015, pp. 501–508.

[17] "sphero robot," http://www.sphero.com/, designed by Sphero.

[18] A. Prabhakar, A. Mavrommati, J. Schultz, and T. D. Murphey, "Autonomous visual rendering using physical motion," in *Workshop on the Algorithmic Foundations of Robotics*, 2016.

[19] J. A. Broderick, D. M. Tilbury, and E. M. Atkins, "Optimal coverage trajectories for a UGV with tradeoffs for energy and time," *Autonomous Robots*, vol. 36, no. 3, pp. 257–271, 2014.

[20] M. Garzón, J. Valente, J. J. Roldán, L. Cancar, A. Barrientos, and J. Del Cerro, "A multirobot system for distributed area coverage and signal searching in large outdoor scenarios," *Journal of Field Robotics*, vol. 33, no. 8, pp. 1087–1106, 2016.

[21] X. Tan, "Autonomous robotic fish as mobile sensor platforms: Challenges and potential solutions," *Marine Technology Society Journal*, vol. 45, no. 4, pp. 31–40, 2011.

[22] M. Castano, A. Mavrommati, T. D. Murphey, and X. Tan, "Trajectory planning and tracking of robotic fish using ergodic exploration," in *American Control Conference*, 2017, pp. 5476–5481.

[23] M. Quigley, M. A. Goodrich, S. Griffiths, A. Eldredge, and R. W. Beard, "Target acquisition, localization, and surveillance using a fixed-wing mini-UAV and gimbaled camera," in *IEEE international conference on robotics and automation*, 2005, pp. 2600–2605.

[24] K. Dogancay, "UAV path planning for passive emitter localization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 2, 2012.

[25] Z. Tang and U. Ozguner, "Motion planning for multitarget surveillance with mobile sensor agents," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 898–908, 2005.

[26] T. H. Summers, M. R. Akella, and M. J. Mears, "Coordinated standoff tracking of moving targets: Control laws and information architectures," *Journal of Guidance, Control, and Dynamics*, vol. 32, no. 1, pp. 56–69, 2009.

[27] V. P. Jilkov and X. R. Li, "On fusion of multiple objectives for UAV search & track path optimization." *Journal of Advances in Information Fusion*, vol. 4, no. 1, pp. 27–39, 2009.

[28] R. R. Pitre, X. R. Li, and R. Delbalzo, "UAV route planning for joint search and track missions – An information-value approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 3, pp. 2551–2565, 2012.

[29] L. C. Pimenta, M. Schwager, Q. Lindsey, V. Kumar, D. Rus, R. C. Mesquita, and G. A. Pereira, "Simultaneous coverage and tracking (SCAT) of moving targets with robot networks," in *Algorithmic foundation of robotics VIII*. Springer, 2009, pp. 85–99.

[30] C. Y. Wong, G. Seet, and S. K. Sim, "Multiple-robot systems for USAR: key design attributes and deployment issues," *International Journal of Advanced Robotic Systems*, vol. 8, no. 1, pp. 85–101, 2011.

[31] Y. Liu and G. Nejat, "Robotic urban search and rescue: A survey from the control perspective," *Journal of Intelligent & Robotic Systems*, vol. 72, no. 2, pp. 147–165, 2013.

[32] A. Barrientos, J. Colorado, J. d. Cerro, A. Martinez, C. Rossi, D. Sanz, and J. Valente, "Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots," *Journal of Field Robotics*, vol. 28, no. 5, pp. 667–689, 2011.

[33] P. A. Plonski, P. Tokekar, and V. Isler, "Energy-efficient path planning for solar-powered mobile robots," *Journal of Field Robotics*, vol. 30, no. 4, pp. 583–601, 2013.

[34] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, "Efficient informative sensing using multiple robots," *Journal of Artificial Intelligence Research*, vol. 34, pp. 707–755, 2009.

[35] E. Galceran and M. Carreras, "A survey on coverage path planning for robotics," *Robotics and Autonomous Systems*, vol. 61, no. 12, pp. 1258–1276, 2013.

[36] C. Luo and S. X. Yang, "A bioinspired neural network for real-time concurrent map building and complete coverage robot navigation in unknown environments," *IEEE Transactions on Neural Networks*, vol. 19, no. 7, pp. 1279–1298, 2008.

[37] H. H. Viet, V.-H. Dang, M. N. U. Laskar, and T. Chung, "BA*: An online complete coverage algorithm for cleaning robots," *Applied intelligence*, vol. 39, no. 2, pp. 217–235, 2013.

[38] N. K. Yilmaz, C. Evangelinos, P. F. Lermusiaux, and N. M. Patrikalakis, "Path planning of autonomous underwater vehicles for adaptive sampling using mixed integer linear programming," *IEEE Journal of Oceanic Engineering*, vol. 33, no. 4, pp. 522–537, 2008.

[39] L. Paull, S. Saeedi, M. Seto, and H. Li, "Sensor-driven online coverage planning for autonomous underwater vehicles," *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 6, pp. 1827–1838, 2013.

[40] E. U. Acar, H. Choset, Y. Zhang, and M. Schervish, "Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods," *The International Journal of Robotics Research*, vol. 22, no. 7-8, pp. 441–466, 2003.

[41] C. W. Bac, E. J. Henten, J. Hemming, and Y. Edan, "Harvesting robots for high-value crops: State-of-the-art review and challenges ahead," *Journal of Field Robotics*, vol. 31, no. 6, pp. 888–911, 2014.

[42] S. L. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410–426, 2012.

[43] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial informatics*, vol. 9, no. 1, pp. 427–438, 2013.

[44] W. Ren, R. W. Beard, and T. W. McLain, "Coordination variables and consensus building in multiple vehicle systems," in *Lecture Notes in Control and Information Sciences*. New York: Springer-Verlag, 2004, vol. 309, pp. 171–188.

[45] J. Cortes, S. Martinez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 11, no. 4, pp. 691–719, 2005.

[46] Y. Kantaros, M. Thanou, and A. Tzes, "Distributed coverage control for concave areas by a heterogeneous Robot–Swarm with visibility sensing constraints," *Automatica*, vol. 53, pp. 195–207, 2015.

[47] S. Rutishauser, N. Correll, and A. Martinoli, "Collaborative coverage using a swarm of networked miniature robots," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 517–525, 2009.

[48] A. Kwok and S. Martínez, "A distributed deterministic annealing algorithm for limited-range sensor coverage," *IEEE Transactions on Control Systems Technology*, vol. 19, no. 4, pp. 792–804, 2011.

[49] M. Zhu and S. Martínez, "Distributed coverage games for energy-aware mobile sensor networks," *SIAM Journal on Control and Optimization*, vol. 51, no. 1, pp. 1–27, 2013.

[50] A. Y. Yazicioglu, M. Egerstedt, and J. S. Shamma, "Communication-free distributed coverage for networked systems," *IEEE Transactions on Control of Network Systems*, vol. PP, no. 99, pp. 1–1, 2016.

[51] D. A. Shell and M. J. Matarić, "Ergodic dynamics for large-scale distributed robot systems," in *International Conference on Unconventional Computation*. Springer, 2006, pp. 254–266.

[52] B. J. Julian, M. Angermann, M. Schwager, and D. Rus, "Distributed robotic sensor networks: An information-theoretic approach," *The International Journal of Robotics Research*, vol. 31, no. 10, pp. 1134–1154, 2012.

[53] A. D. Wilson, J. A. Schultz, A. R. Ansari, and T. D. Murphey, "Dynamic task execution using active parameter identification with the BAXTER research robot," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 1, pp. 391–397, 2017.

[54] C. Leung, S. Huang, G. Dissanayake, and T. Furukawa, "Trajectory planning for multiple robots in bearing-only target localisation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 3978–3983.

[55] X. Lan and M. Schwager, "Rapidly exploring random cycles: Persistent estimation of spatiotemporal fields with multiple sensing robots," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1230–1244, 2016.

[56] J. P. Helferty and D. R. Mudgett, "Optimal observer trajectories for bearings only tracking by minimizing the trace of the cramer-rao lower bound," in *IEEE Conference on Decision and Control*, 1993, pp. 936–939.

[57] N. Cao, K. H. Low, and J. M. Dolan, "Multi-robot informative path planning for active sensing of environmental phenomena: A tale of two algorithms," in *International Conference on Autonomous Agents and Multi-Agent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2013, pp. 7–14.

[58] P. Dames, P. Tokekar, and V. Kumar, "Detecting, localizing, and tracking an unknown number of moving targets using a team of mobile robots," *International Symposium on Robotics Research*, 2015.

[59] P. Dames and V. Kumar, "Autonomous localization of an unknown number of targets without data association using teams of mobile sensors," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 850–864, 2015.

[60] H. Yu, K. Meier, M. Argyle, and R. W. Beard, "Cooperative path planning for target tracking in urban environments using unmanned air and ground vehicles," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 541–552, 2015.

[61] S. Kim, H. Oh, and A. Tsourdos, "Nonlinear model predictive coordinated standoff tracking of a moving ground vehicle," *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 2, pp. 557–566, 2013.

[62] R. Anderson and D. Milutinović, "A stochastic approach to Dubins feedback control for target tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 3917–3922.

[63] J. R. Spletzer and C. J. Taylor, "Dynamic sensor planning and control for optimally tracking targets," *The International Journal of Robotics Research*, vol. 22, no. 1, pp. 7–20, 2003.

[64] S. Zhu, D. Wang, and C. B. Low, "Ground target tracking using UAV with input constraints," *Journal of Intelligent & Robotic Systems*, vol. 69, no. 1-4, pp. 417–429, 2013.

[65] U. Zengin and A. Dogan, "Real-time target tracking for autonomous UAVs in adversarial environments: A gradient search algorithm," *IEEE Transactions on Robotics*, vol. 23, no. 2, pp. 294–307, 2007.

[66] P. Yao, H. Wang, and Z. Su, "Real-time path planning of unmanned aerial vehicle for target tracking and obstacle avoidance in complex dynamic environment," *Aerospace Science and Technology*, vol. 47, pp. 269–279, 2015.

[67] P. Tokekar, J. Vander Hook, and V. Isler, "Active target localization for bearing based robotic telemetry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 488–493.

[68] S. M. Ross, R. G. Cobb, and W. P. Baker, "Stochastic real-time optimal control for bearing-only trajectory planning," *International Journal of Micro Air Vehicles*, vol. 6, no. 1, pp. 1–27, 2014.

[69] D. B. Barber, J. D. Redding, T. W. McLain, R. W. Beard, and C. N. Taylor, "Vision-based target geo-location using a fixed-wing miniature air vehicle," *Journal of Intelligent and Robotic Systems*, vol. 47, no. 4, pp. 361–382, 2006.

[70] L. Ma and N. Hovakimyan, "Cooperative target tracking in balanced circular formation: Multiple UAVs tracking a ground vehicle," in *American Control Conference*, 2013, pp. 5386–5391.

[71] S. A. Quintero, M. Ludkovski, and J. P. Hespanha, "Stochastic optimal coordination of small UAVs for target tracking using regression-based dynamic programming," *Journal of Intelligent & Robotic Systems*, vol. 82, no. 1, pp. 135–162, 2016.

[72] G. Gu, P. Chandler, C. Schumacher, A. Sparks, and M. Pachter, "Optimal cooperative sensing using a team of UAVs," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 42, no. 4, pp. 1446–1458, 2006.

[73] G. Mathew and I. Mezić, "Metrics for ergodicity and design of ergodic dynamics for multi-agent systems," *Physica D: Nonlinear Phenomena*, vol. 240, no. 4, pp. 432–442, 2011.

[74] D. E. Kirk, *Optimal control theory: An introduction*. Courier Corporation, 2012.

[75] L. M. Miller and T. D. Murphey, "Trajectory optimization for continuous ergodic exploration," in *American Control Conference*, 2013, pp. 4196–4201.

[76] G. D. L. Torre, K. Flaßkamp, A. Prabhakar, and T. D. Murphey, "Ergodic exploration with stochastic sensor dynamics," in *American Control Conference*, 2016, pp. 2971–2976.

[77] D. Liberzon, *Calculus of variations and optimal control theory: A concise introduction*. Princeton University Press, 2012.

[78] F. A. Fontes, "A general framework to design stabilizing nonlinear model predictive controllers," *Systems & Control Letters*, vol. 42, no. 2, pp. 127–143, 2001.

[79] A. R. Ansari and T. D. Murphey, "Sequential Action Control: Closed-form optimal control for nonlinear and nonsmooth systems," *IEEE Transactions on Robotics*, vol. 32, no. 5, pp. 1196–1214, 2016.

[80] E. Tzorakoleftherakis, A. R. Ansari, A. Wilson, J. Schultz, and T. D. Murphey, "Model-based reactive control for hybrid and high-dimensional robotic systems," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 431–438, 2016.

[81] D. Q. Mayne, "Model predictive control: Recent developments and future promise," *Automatica*, vol. 50, no. 12, pp. 2967–2986, 2014.

[82] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, no. 1, pp. 25–57, 2006.

[83] M. Egerstedt, Y. Wardi, and H. Axelsson, "Transition-time optimization for switched-mode dynamical systems," *IEEE Transactions on Automatic Control*, vol. 51, no. 1, pp. 110–115, 2006.

[84] T. M. Caldwell and T. D. Murphey, "Projection-based iterative mode scheduling for switched systems," *Nonlinear Analysis: Hybrid Systems*, vol. 21, pp. 59–83, 2016.

[85] S. L. de Oliveira Kothare and M. Morari, "Contractive model predictive control for constrained nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 45, no. 6, pp. 1053–1071, 2000.

[86] E. Camponogara, D. Jia, B. H. Krogh, and S. Talukdar, "Distributed model predictive control," *IEEE Control Systems*, vol. 22, no. 1, pp. 44–52, 2002.

[87] F. Xie and R. Fierro, "First-state contractive model predictive control of nonholonomic mobile robots," in *American Control Conference*, 2008, pp. 3494–3499.

[88] G. Ferrari-Trecate, L. Galbusera, M. P. E. Marciandi, and R. Scattolini, "Model predictive control schemes for consensus in multi-agent systems with single-and double-integrator dynamics," *IEEE Transactions on Automatic Control*, vol. 54, no. 11, pp. 2560–2572, 2009.

[89] L. Grüne and J. Pannek, *Nonlinear model predictive control*. Springer, 2011.

[90] J. H. Lee, "Model predictive control: Review of the three decades of development," *International Journal of Control, Automation and Systems*, vol. 9, no. 3, pp. 415–424, 2011.

[91] I. Barbalat, "Systemes déquations différentielles doscillations non linéaires," *Rev. Math. Pures Appl*, vol. 4, no. 2, pp. 267–270, 1959.

[92] H. Michalska and R. Vinter, "Nonlinear stabilization using discontinuous moving-horizon control," *IMA Journal of Mathematical Control and Information*, vol. 11, no. 4, pp. 321–340, 1994.

[93] E. Tzorakoleftherakis and T. Murphey, "Iterative sequential action control for stable, model-based control of nonlinear systems," *arXiv preprint arXiv:1706.08932*, 2017.

[94] A. Emery and A. V. Nenarokomov, "Optimal experiment design," *Measurement Science and Technology*, vol. 9, no. 6, p. 864, 1998.

[95] B. R. Frieden, *Science from Fisher information: A unification*. Cambridge University Press, 2004.

[96] D. Ucinski, *Optimal measurement methods for distributed parameter system identification*. CRC Press, 2004.

[97] L. R. G. Carrillo, A. E. D. López, R. Lozano, and C. Pégard, "Modeling the quad-rotor mini-rotorcraft," in *Quad Rotorcraft Control*. Springer, 2013, pp. 23–34.

[98] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[99] T. Luukkonen, "Modelling and control of quadcopter," *Independent research project in applied mathematics, Espoo*, 2011.

[100] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[101] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," in *AeroSense'97*. International Society for Optics and Photonics, 1997, pp. 182–193.

[102] E. S. Jones and S. Soatto, "Visual-inertial navigation, mapping and localization: A scalable real-time causal approach," *The International Journal of Robotics Research*, vol. 30, no. 4, pp. 407–430, 2011.

[103] V. J. Aidala, "Kalman filter behavior in bearings-only tracking applications," *IEEE Transactions on Aerospace and Electronic Systems*, no. 1, pp. 29–39, 1979.

[104] T. Lee and S. Soatto, "Learning and matching multiscale template descriptors for real-time detection, localization and tracking," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1457–1464.

[105] F. Zhang and N. E. Leonard, "Cooperative filters and control for cooperative exploration," *IEEE Transactions on Automatic Control*, vol. 55, no. 3, pp. 650–663, 2010.

[106] (2014) Robot Operating System. Willow Garage. [Online]. Available: http://www.ros.org/

Todd D. Murphey received his B.S. degree in mathematics from the University of Arizona and the Ph.D. degree in Control and Dynamical Systems from the California Institute of Technology.

He is a Professor of Mechanical Engineering at Northwestern University. His laboratory is part of the Neuroscience and Robotics Laboratory, and his research interests include robotics, control, computational methods for biomechanical systems, and computational neuroscience.

Honors include the National Science Foundation CAREER award in 2006, membership in the 2014-2015 DARPA/IDA Defense Science Study Group, and Northwestern's Professorship of Teaching Excellence. He is a Senior Editor of the IEEE Transactions on Robotics.

Anastasia Mavrommati received the B.S. and M.S. degrees in electrical and computer engineering from the University of Patras, Patras, Greece in 2012, and the M.S. and Ph.D. degrees in mechanical engineering from Northwestern University, Evanston, IL, USA, in 2015 and 2017 respectively. Her Ph.D. thesis focused on real-time algorithms for information acquisition and hybrid control. She is currently a Research Scientist at Schlumberger-Doll Research, Cambridge, MA, conducting research on robotics applications for the energy industry.
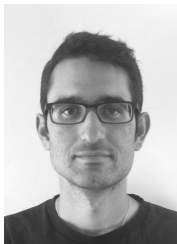
Emmanouil Tzorakoleftherakis received the joint B.S./M.S. degree in electrical and computer engineering from University of Patras, Greece in 2012, and the M.S. and Ph.D. degrees in mechanical engineering from Northwestern University, Evanston, IL, USA, in 2015 and 2017 respectively.

Ian Abraham received the B.S. degree in mechanical engineering from Rutgers University, New Brunswick, NJ, USA, in 2014.

He is currently a Ph.D candidate in the Neuroscience and Robotics Laboratory at Northwestern University, IL, USA. His research interests include data-driven control, active sensing, and active learning.