A Pareto-Optimal Local Optimization Framework for Multi-Objective Ergodic Search

Zhongqiang Ren[1], Akshaya Kesarimangalam Srinivasan[1], Bhaskar Vundurthy[1], Ian Abraham[2] and Howie Choset[1]

***Abstract*—Our work is motivated by humanitarian assistant and disaster relief (HADR) where often it is critical to find signs of life in the presence of conflicting criteria, objectives, and information. We believe ergodic search can provide a framework for exploiting available information as well as exploring for new information in applications such as HADR. Existing ergodic search methods typically consider search using only a single information map. However, one can readily envision many scenarios where multiple information maps that encode different types of relevant information are used. Ergodic search methods currently do not possess the ability to simultaneously search multiple information maps, nor do they have a way to balance which information gets priority. This leads us to formulate a Multi-Objective Ergodic Search (MO-ES) problem, which aims to find the so-called Pareto-optimal solutions, for the purpose of providing human decision makers various solutions that trade off among conflicting criteria. To efficiently solve MO-ES, we develop a framework called Sequential Local Ergodic Search (SL-ES), which leverages the recent advances in ergodic search methods as well as the idea of local optimization to efficiently compute Pareto-optimal solutions. Our numerical results show that SL-ES computes solutions of better quality and runs faster than the baselines.**

***Index Terms*—Motion and Path Planning, Optimization and Optimal Control, Ergodic Search**

I. Introduction

This paper considers a trajectory planning problem for area search, which arises in applications such as search and rescue [1], [2], environment monitoring [3], target localization [4], [5]. Given an information map (hereafter abbreviated as info map), which describes the prior knowledge in form of a distribution over the area to be searched, the problem requires planning a trajectory to efficiently gather information. Common approaches to this problem span a spectrum from spatial decomposition methods [6]–[8], which uniformly cover the area, to information-theoretic approaches [3], [9], which greedily move the robot to the next location with the highest information gain. This paper is interested in the middle of the spectrum with a type of search called ergodic search [10]–[12]. Ergodic search optimizes an ergodic metric to plan trajectories along which the time spent in a region is proportional to the amount of information in that region. Ergodic search inherently balances exploitation (i.e., myopically searching high-information areas) and exploration (i.e., attempting to visit all possible locations for new information), and is thus able to intelligently determine the robot motion to collect information in the long term.

Existing ergodic search algorithms [10]–[12] consider covering only a single info map. However, one can envision scenarios where multiple different info maps, each of which encodes one type of information, may need to be searched simultaneously. Each info map corresponds to an objective to be optimized and hence multi-objective optimization. As an example, consider a hazardous material warehouse with leakage where a robot is deployed to search for both survivors and leakage sources (Fig. 1 (a)). Multiple info maps describing probable locations of survivors and leakage sources are required to be simultaneously covered. This problem is truly multi-objective in the sense that multiple info maps cannot be combined, say as a linear combination of info maps, as the weights, and hence their relative importance, is not known.

In this paper, we formulate a Multi-Objective Ergodic Search (MO-ES) problem, whose solutions are trajectories that can simultaneously cover multiple info maps. In general, there is no single trajectory that optimizes the ergodic metrics with regard to all info maps at the same time. Thus, this paper seeks to find a set of Pareto-optimal solutions (trajectories): a solution is Pareto-optimal if one can not improve the ergodic metric with respect to one info map without deteriorating the ergodic metric with regard to at least one of the other info maps. We believe the visualization of a set of Pareto-optimal solutions can help the human decision makers (who are often involved in the task [2], [13]) make more informed decisions based on their domain knowledge.

Existing approaches that can be used to solve MO-ES include general-purpose multi-objective genetic algorithms (MOGA) [14]–[16]. While being applicable to various problems, MOGAs typically fail to leverage the underlying structure of MO-ES problems (such as the dynamics of the robot and local metric structures e.g., convexity), which can make them inefficient to optimize. Another existing approach is the scalarization method [15], [17], which can be applied to solve MO-ES by sampling a set of weight vectors, computing the weighted-sum of the objectives for each weight vector, and running a single-objective algorithm to optimize the scalarized objective function. While being able to leverage the existing single-objective algorithms, the scalarization method can be time-consuming as it optimizes for each weight vector in order to obtain a set of Pareto-optimal solutions.

This paper develops a framework called Sequential Local Ergodic Search (SL-ES) to quickly obtain a set of Pareto-

Manuscript received September, 30, 2022; Revised February, 10, 2023; Accepted May, 31, 2023. *(Corresponding author: Zhongqiang Ren.)*

[1] Zhongqiang Ren, Akshaya Kesarimangalam Srinivasan, Bhaskar Vundurthy and Howie Choset are with Carnegie Mellon University, 5000 Forbes Ave., Pittsburgh, PA 15213, USA. (email: {zhongqir, akesarim, pvundurt, choset}@andrew.cmu.edu).

[2]Ian Abraham is with Yale University, 17 Hillhouse Avenue, New Haven, CT 06511, USA. (email: ian.abraham@yale.edu)

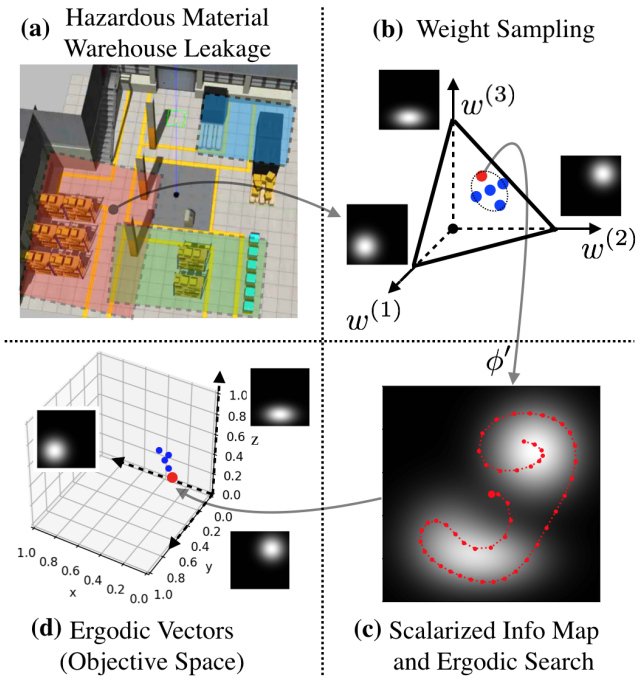


Fig. 1. A visualization of the MO-ES problem and our method. (a) shows a hazardous material warehouse with leakage, where colored areas indicate different types of information/targets such as survivors, leakage sources, etc. Each type of information is represented as an info map. (b) shows the weight space $\mathcal{B}$ in the presence of three objectives, where $w^{(i)}$ is the relative weight of the corresponding info map $\phi^{(i)}$, with $i = 1, 2, 3$. (c) shows the scalarized info map ϕ', which is the weighted-sum of all three info maps. An ergodic trajectory is planned with respect to ϕ'. (d) shows the objective space, where each element is an ergodic vector that describes the ergodic metric of the computed trajectory with respect to $\phi^{(1)}, \phi^{(2)}, \phi^{(3)}$. The computed ergodic vectors are guaranteed to be Pareto-optimal.

optimal solutions, and the framework is conceptually visualized in Fig. 1. First, SL-ES uses a set of weight vectors, and computes a *scalarized info map* by taking the weighted-sum of the info maps to be covered using each weight vector. The idea of scalarizing info maps (rather than objective functions) allows us to leverage the existing various (single-objective) ergodic search algorithms. We formally prove that optimizing the ergodic metric of a trajectory with respect to a scalarized info map yields a trajectory that is guaranteed to be Pareto-optimal. Furthermore, SL-ES leverages the idea of local optimization based on the inherent convexity of the ergodic metric in the Fourier coefficient space. SL-ES samples weight vectors from the weight space (i.e., the space that contains all possible weight vectors) in a breadth-first manner by (i) episodically sampling new weight vectors in the neighborhood of the current weight vector, and (ii) optimizing the trajectory corresponding to the new weight vector by using the current solution as the initial guess (to warm-start the optimization). Finally, to expedite the computation, we also develop a variant called Adaptive SL-ES (A-SL-ES), which can adjust the density of the sampled weight vectors based on the *similarity* of the info maps in the Fourier coefficient space. Our numerical results show that SL-ES and A-SL-ES compute a set of solution trajectories with better ergodic metrics than applying MOGAs to MO-ES. Additionally, SL-ES and A-SL-ES require less than half of the run time of a naive scalarization method that does not leverage local optimization. We simulate our method in a hazardous material warehouse in ROS/Gazebo, and verify that the planned trajectory can be executed on physical robots.

The prior version of this work has appeared in [18]. Different from [18], this paper generalizes the method into a framework with solution quality guarantees by (i) proving the Pareto-optimality of the computed solutions by SL-ES and A-SL-ES, (ii) extending the method from single-agent to multi-agent, and (iii) providing more numerical results in various info maps, discussion and physical robot demonstrations. The rest of this paper is organized as follows. Sec. I-A reviews related work and Sec. II introduces basic concepts and the problem definition. We then elaborate the method in Sec. III and prove the property of the method in Sec. IV. Finally, we discuss the results in Sec. V and conclude in Sec. VI.

A. Other Related Work

1) Ergodic Coverage: A trajectory is ergodic with respect to an info map if the amount of time spent in a region is proportional to the amount of information in that region. Ergodic metrics, such as [10], measure how far a trajectory is from being ergodic, and by iteratively minimizing the metric, an ergodic trajectory can be computed. Ergodic trajectory planning has been investigated within the framework of receding horizon control [11], stochastic optimization [19], and has been leveraged for active learning and search [12], [20], [21], decentralized exploration [22], real-time area coverage and target localization [23], etc. However, we are not aware of any ergodic search method that considers covering multiple info maps at the same time, which is the focus of this paper.

2) Multi-Objective Optimization: Multi-Objective Optimization (MOO) is a broad topic [15], [17] and has been investigated in robotics-related problems such as path planning [24], [25], reinforcement learning [26], design [27] and multi-agent systems [28]. With respect to MOO for search tasks, existing work has considered simultaneously optimizing exploration and exploitation for environment monitoring tasks [3]. In [1], a trajectory is planned to cover a single info map using the ergodic metric, while other "non-ergodic" objectives are considered by using ϵ-constraints. Different from [1], this paper aims to plan ergodic trajectories to cover multiple info maps, where each info map corresponds to an objective.

II. Preliminaries

A. Ergodic Metric

Let $\mathcal{W} = [0, L_1] \times [0, L_2] \times \cdots \times [0, L_\nu] \subset \mathbb{R}^\nu, \nu \in \{2, 3\}$ denote a ν-dimensional workspace that is to be explored by the robot. The robot has an n-dimensional state space ($n \geq \nu$), and let $q_n : [0, T] \to \mathbb{R}^n$ denote a trajectory in the state space with $T \in \mathbb{R}^+$ representing the time horizon. The robot has deterministic dynamics given by $\dot{q}_n(t) = f(q_n(t), u(t))$, where $u(t)$ is the control input of the robot. Additionally, for each trajectory q_n, let $q : [0, T] \to \mathcal{W}$ denote the corresponding trajectory in the workspace (instead of in the state space).

Let $c(x,q), x \in \mathcal{W}$ denote the time-averaged statistics of a trajectory q, which is defined as:

$$c(x,q) = \frac{1}{T}\int_0^T \delta(x - q(\tau))d\tau, \tag{1}$$

where δ is a Dirac function. Let $\phi : \mathcal{W} \to \mathbb{R}$ denote a static info map that describes the amount of information at each location in the workspace. Each info map is a probability distribution with $\int_{\mathcal{W}} \phi = 1$ and $\phi(x) \geq 0, \forall x \in \mathcal{W}$. An ergodic metric [10] between $c(x,q)$ and an info map ϕ is defined as:

$$\begin{aligned}\mathcal{E}(\phi, q) &= \sum_{k=0}^{K} \lambda_k (c_k - \phi_k)^2 \\ &= \sum_{k=0}^{K} \lambda_k \left(\frac{1}{T}\int_0^T F_k(q(\tau))d\tau - \phi_k\right)^2\end{aligned} \tag{2}$$

where (i) $\phi_k = \int_{\mathcal{W}} \phi(x) F_k(x) dx$ represents the Fourier coefficients of the info map, with $F_k(q) = \frac{1}{h_k}\Pi_{j=1}^{\nu} \cos(\frac{k_j \pi q_j}{L_j})$ being the cosine basis function for some index $k \in \mathbb{N}^{\nu}$ and K being the number of Fourier bases considered, (ii) c_k denotes the Fourier coefficient of $c(x,q)$, (iii) h_k denotes the normalization factor as defined in [10], and (iv) $\lambda_k = (1 + ||k||^2)^{-\frac{\nu+1}{2}}$ denotes the weight for each corresponding Fourier coefficient.

B. Ergodic Vector and Pareto-Optimality

This paper seeks to plan robot trajectories to simultaneously cover multiple (static) info maps. For clarity, we use the superscript in $\phi^{(i)}$ to denote a specific info map, with $i \in \{1, 2, \cdots, m\}$ where m is a finite number indicating the total number of the info maps to be covered. Let $\vec{\mathcal{E}}(q) = (\mathcal{E}(\phi^{(1)}, q), \mathcal{E}(\phi^{(2)}, q), \cdots, \mathcal{E}(\phi^{(m)}, q))$ denote an *ergodic vector*, which describes the ergodic metrics of the trajectory q with respect to all info maps. Each component of $\vec{\mathcal{E}}(q)$ corresponds to an objective function to be minimized. To compare any two trajectories, we compare the ergodic vectors corresponding to them using the dominance relation from the multi-objective optimization literature.

Definition 1 (Dominance [17])**.** *Given two vectors a and b of length m, a dominates b, notationally $a \succeq b$, if and only if $a(j) \leq b(j)$, $\forall j \in \{1, 2, \cdots, m\}$ and $a(j) < b(j)$, $\exists j \in \{1, 2, \cdots, m\}$.*

If a does not dominate b, this non-dominance is denoted as $a \not\succeq b$. Given two trajectories q_1, q_2 (with the same time horizon $[0, T]$), we say q_1 dominates q_2 (denoted as $q_1 \succeq q_2$) if $\vec{\mathcal{E}}(q_1) \succeq \vec{\mathcal{E}}(q_2)$. Any two trajectories are non-dominated to each other if the corresponding ergodic vectors do not dominate each other. Among all feasible trajectories, the set of all non-dominated trajectories is called the *Pareto-optimal* (solution) set, and the set of the corresponding ergodic vectors is called the Pareto-optimal front.

C. Problem Statement

This paper considers the following *Multi-Objective Ergodic Search (MO-ES)* problem. Given a set of info maps and the dynamics of the robot, the goal is to compute a set of dynamically feasible trajectories, whose corresponding ergodic vectors are Pareto-optimal.

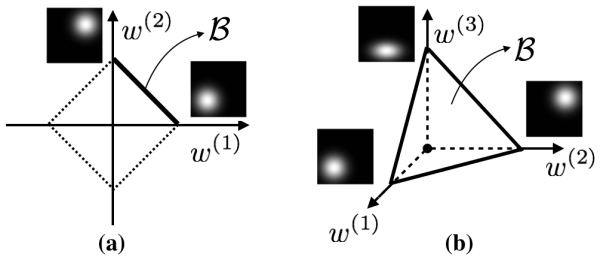


Fig. 2. Examples of the weight space $\mathcal{B}$ when (a) $m = 2$ and (b) $m = 3$. Symbol $w^{(i)}$ stands for the i-th component of a weight vector $\vec{w}$.

Remark 1. *We first focus on the single-agent version of the problem in an obstacle-free workspace. We will discuss the extension to the multi-agent setting in Sec. III-G.*

III. Method

A. Basic Concepts and Overview

Let $\mathcal{B} := \{\vec{w}, w^{(i)} > 0, i = 1, 2, \cdots, m, ||\vec{w}||_1 = 1\}$ denote the space of possible weight vectors, which is hereafter referred to as the *weight space*. The weight space is the first quadrant of the m-dimensional ℓ_1-norm unit sphere. For clarity, we use the superscript in $w^{(i)}$ to denote the i-th component of the weight vector $w \in \mathcal{B}$, which is consistent with the superscript notation in $\phi^{(i)}$. Examples of $\mathcal{B}$ when $m = 2, 3$ are shown in Fig. 2. An info map ϕ can be decomposed with respect to (abbreviated as w.r.t.) a set of Fourier bases as $\phi = \Sigma_{k=0}^{K} \phi_k F_k$, where ϕ_k denotes the Fourier coefficient corresponding to each Fourier basis function $F_k, k \in \{0, 1, 2, \cdots, K\}$. In practice, K is often selected to be a finite number instead of infinity.

Given a weight vector $\vec{w} \in \mathcal{B}$, we choose to scalarize the info maps, as opposed to scalarizing the objective functions $\vec{\mathcal{E}}(q)$, by taking the weighted sum of the info maps to be covered. By doing so, the existing ergodic search methods that consider a single info map can be leveraged. We will discuss this in detail later. This *scalarized info map* can be represented as the weighted-sum of the corresponding Fourier coefficients:

$$\begin{aligned}\phi' &= \sum_{k=0}^{K} \phi'_k F_k \\ &= \sum_{i=1}^{m} w^{(i)} \phi^{(i)} = \sum_{i=1}^{m} w^{(i)} \left(\sum_{k=0}^{K} \phi_k^{(i)} F_k\right).\end{aligned} \tag{3}$$

For each $k \in \{0, 1, \cdots, K\}$:

$$\phi'_k = \sum_{i=1}^{m} w^{(i)} \phi_k^{(i)} = \vec{w} \cdot \Phi_k, \tag{4}$$

where $\Phi_k = (\phi_k^{(1)}, \phi_k^{(2)}, \cdots, \phi_k^{(m)})$, $\vec{w} = (w^{(1)}, w^{(2)}, \cdots, w^{(m)})$, and $\cdot$ stands for the dot product.

The ergodic metric of q, whose time averaged statistics is described as a set of Fourier coefficients c_k, w.r.t. ϕ' is

$$\mathcal{E}(\phi', q) = \sum_{k=0}^{K} \lambda_k (c_k - \phi'_k)^2 = \sum_{k=0}^{K} \lambda_k (c_k - \vec{w} \cdot \Phi_k)^2 \quad (5)$$

To obtain a set of Pareto-optimal trajectories, this paper develops a framework (Fig. 1): intuitively, in each planning episode, a $\vec{w}$ is sampled from $\mathcal{B}$ and a corresponding scalarized info map ϕ' is computed with (4). Then, an ergodic trajectory w.r.t. ϕ' is planned by minimizing $\mathcal{E}(\phi', q)$ in (5). Note that, instead of using the weight vector $\vec{w}$ to scalarize the ergodic vector (i.e., the objective vector) $\vec{\mathcal{E}}$ as introduced in Sec. II-B, our framework scalarizes the info maps, which allows us to leverage the existing (single-objective) ergodic search algorithms to cover the scalarized info map. We will prove in Sec. IV that the trajectories computed by our framework are Pareto-optimal.

Also note that, with a given initial state of the robot $q(0)$ and a control $u(t)$, a unique trajectory can be specified (via the so-called forward simulation). Thus, for presentation purposes, we use a control $u(t), t \in [0, T]$ to identify a trajectory, and let $u(t)|_{\vec{w}}$ denote the ergodic trajectory computed w.r.t. the scalarized info map based on $\vec{w}$. In other words, for each $\vec{w} \in \mathcal{B}$, a corresponding ergodic trajectory $u(t)|_{\vec{w}}$ can be computed.

Additionally, with (5), we can observe that:

- $\mathcal{E}(\phi', q)$ (i.e., the objective function to be minimized after scalarizing the info maps) is a *convex* function w.r.t. $\vec{w}$ and c_k.
- Although c_k is non-convex with respect to $u(t)$ due to the robot dynamics and the Fourier bases, existing ergodic search algorithms [10], [12], [23] have shown that this non-convexity can be handled by iterative gradient descent optimization in practice.

Based on these observations, we take the view that a set of trajectories can be efficiently obtained by episodically sampling new $\vec{w}$ in the neighborhood of the current weight vector, and running local optimization in each episode. Following this idea, we propose a framework called Sequential Local Ergodic Search (SL-ES), which is explained in the next section.

B. Sequential Local Ergodic Search (SL-ES)

Intuitively, SL-ES covers (or say explores) the weight space $\mathcal{B}$ from some initial weight vector $\vec{w}_{init}$ in a breadth-first manner in order to compute a set of Pareto-optimal solutions. SL-ES iteratively (i) scalarizes the info maps based on the current weight vector $\vec{w}$, (ii) leverages regular (single-objective) ergodic search to compute a trajectory (represented by $u(t)|_{\vec{w}}$), and (iii) samples new weight vectors $\vec{w}'$ from $\mathcal{B}$ in the neighborhood of $\vec{w}$ and uses $u(t)|_{\vec{w}}$ as an initial guess to optimize the ergodic trajectory corresponding to $\vec{w}'$. The above process iterates until $\mathcal{B}$ has been fully explored by the sampled weight vectors.

Specifically, as shown in Alg. 1, SL-ES begins by initializing a weight vector $\vec{w}_{init}$ (Line 1), which can be either randomly sampled from $\mathcal{B}$, or specified by the user based on the domain knowledge of the specific application. An initial control $u_{init}(t)|_{\vec{w}_{init}}$ corresponding to $\vec{w}$ is also initialized (Line 2), which will later be used as the initial guess for the first episode of the ergodic search. Let OPEN denote a first-in-first-out queue containing candidate weight vectors that need expansion, and expanding a weight vector $\vec{w}$ means computing $u(t)|_{\vec{w}}$ and sampling new weight vectors in the neighborhood of $\vec{w}$. Let CLOSED denote a set of weight vectors that have been expanded, and let $\mathcal{S}$ denote the set of corresponding $u(t)|_{\vec{w}}$ for each $\vec{w} \in$ CLOSED that have been computed at any time during the computation. Initially, CLOSED and $\mathcal{S}$ are all initialized as empty sets (Line 3) and OPEN is initialized by adding $\vec{w}_{init}$ (Line 4).

Algorithm 1 Pseudocode for SL-ES

1: $\vec{w}_{init} \leftarrow$ *InitWeight*()
2: $u_{init}(t)|_{\vec{w}_{init}} = 0$
3: CLOSED $\leftarrow \emptyset$, $\mathcal{S} \leftarrow \emptyset$
4: OPEN $\leftarrow \{\vec{w}_{init}\}$
5: **while** OPEN is not empty **do**
6: $\quad \vec{w} \leftarrow$ OPEN.pop()
7: $\quad$ Compute $\{\phi'_k, \forall k \in \{0, 1 \cdots, K\}\}$ with $\vec{w}$ and (4)
8: $\quad u(t)|_{\vec{w}} \leftarrow$ *ErgodicSearch*($\{\phi'_k\}$, $u_{init}(t)|_{\vec{w}}$)
9: $\quad$ Add $\vec{w}$ into CLOSED
10: $\quad$ Add $u(t)|_{\vec{w}}$ into $\mathcal{S}$
11: $\quad$ **for all** $\vec{w}' \in$ *Neighbor*($\vec{w}$) **do**
12: $\quad\quad$ **if** $\vec{w}' \notin$ OPEN $\cup$ CLOSED **then**
13: $\quad\quad\quad u_{init}(t)|_{\vec{w}'} \leftarrow u(t)|_{\vec{w}}$
14: $\quad\quad\quad$ OPEN.push($\vec{w}'$)
15: **return** $\mathcal{S}$ and the corresponding ergodic vectors

In each planning *episode* (Lines 5-14), a weight vector $\vec{w}$ is popped from OPEN and the corresponding scalarized info map ϕ' is computed based on (4), which is represented by its Fourier coefficients (Line 7). Then, a regular ergodic search algorithm is invoked to cover ϕ', which iteratively minimizes (5) from the initial guess $u_{init}(t)|_{\vec{w}}$ (see Sec. III-C). The computed solution trajectory (represented by $u(t)|_{\vec{w}}$) as well as the corresponding $\vec{w}$ are then added to $\mathcal{S}$ and CLOSED respectively. Finally, neighbor weight vectors of $\vec{w}$ in $\mathcal{B}$ are sampled (see Sec. III-D and III-E) and is represented by *Neighbor*($\vec{w}$). For each $\vec{w}' \in$ *Neighbor*($\vec{w}$), if $\vec{w}'$ has not been generated yet (i.e., $\vec{w}' \notin$ OPEN $\cup$ CLOSED), $\vec{w}'$ is added to OPEN for future expansion.

SL-ES terminates when OPEN is empty, which indicates that $\mathcal{B}$ has been fully covered by the sampled weight vectors. At termination, $\mathcal{S}$ is returned (Line 15), which contains a set of control trajectories along with the ergodic vectors corresponding to these trajectories, which are Pareto-optimal.

C. Ergodic Search Procedure

A benefit of SL-ES is its ability to leverage existing ergodic search algorithms to cover ϕ' in procedure *ErgodicSearch* within each planning episode. This paper leverages the existing approach in [22], which iteratively minimizes the ergodic metric as introduced in (5) within an optimal control framework, and is able to handle general non-linear dynamics of the robot. Other ergodic planners, such as [10], [23], [29], can also be used to implement the *ErgodicSearch* procedure

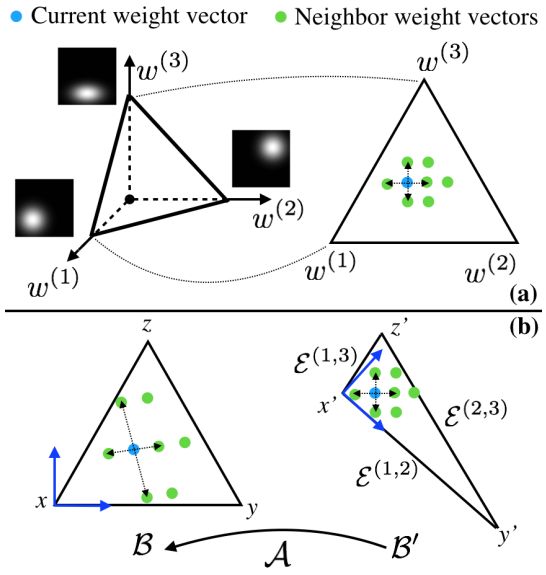


Fig. 3. (a) shows the basic sampling method in the weight space $\mathcal{B}$. (b) shows the adaptive sampling method in the affine transformed weight space $\mathcal{B}'$. Each sampled point in $\mathcal{B}'$ can be affine transformed to a (valid) weight vector in $\mathcal{B}$.

within the framework of SL-ES, in order to handle real-time requirements on computation or obstacle avoidance constraints in the workspace.

In each planning episode, SL-ES invokes *ErgodicSearch* with a specific initial guess $u_{init}|_{\vec{w}}$, instead of using a random or zero control as the initial guess. Specifically, this initial guess is set to be a solution $u(t)|_{\vec{w}'}$ computed in the previous episodes, whose corresponding weight vector $\vec{w}'$ is close to the current weight vector $\vec{w}$ in the weight space $\mathcal{B}$. As shown in Sec. V), this "local optimization" strategy often expedites the overall planning in practice in comparison with a naive scalarization method, which uses $u(t) = 0$ as the initial guess for each episode.

D. Basic Version of Neighbor Sampling

While SL-ES is general to arbitrary $m > 1$, to simplify the presentation, we limit our focus to $m = 2, 3$. Given a weight vector $\vec{w} \in \mathcal{B}$, this paper takes a deterministic sampling strategy with a hyper-parameter d denoting the sampling step size. When $m = 2$, $\mathcal{B}$ is a closed line segment, and the neighbors of a given $\vec{w}$ are defined to be the weight vectors that is of distance d away from $\vec{w}$ along the line segment. When $m = 3$, $\mathcal{B} \subset \mathbb{R}^2$ is the closed set enclosed by a triangle as shown in Fig. 2(b). The neighbors of a given $\vec{w}$ are defined to be the four weight vectors that are of distance d away from $\vec{w}$ along the four cardinal directions, as shown in Fig. 3(a).

In general, $\mathcal{B}$ is the first quadrant of the m-dimensional ℓ_1-norm unit sphere, which is an $(m-1)$-dimensional bounded closed set. In other words, $\mathcal{B}$ is an $(m-1)$-simplex, and each corner point of $\mathcal{B}$ corresponds to an info map to be covered. Since $\mathcal{B}$ is bounded, the aforementioned deterministic sampling strategy generates a finite number of weight vectors from $\mathcal{B}$, and SL-ES is guaranteed to terminate when all these sampled weight vectors are expanded. Additionally, this sampling method can be generalized to $m > 3$. However, the total number of possible samples grows exponentially w.r.t. m, and we leave this potential scalability issue (when m is large) to our future work.

A limitation of this deterministic sampling strategy is that it does not consider the *similarity* between info maps to be covered. For example, if two info maps to be covered are similar (or very different) to each other, then only a few (or a lot of) weight vectors are needed in order to obtain a good representation of the Pareto-optimal front. We handle this limitation in the ensuing section.

E. Adaptive Neighbor Sampling

This section develops an *adaptive* neighbor sampling method, which can adjust the density of samples based on the similarity of info maps to be covered. Let $\mathcal{E}^{(i,j)}$ denote a metric of similarity between two info maps $\phi^{(i)}, \phi^{(j)}$ in the Fourier coefficient space, which resembles the ergodic metric between a trajectory and an info map [10]:

$$\mathcal{E}^{(i,j)} = \sqrt{\sum_{k=0}^{K} \lambda_k (\phi_k^{(i)} - \phi_k^{(j)})^2}. \tag{6}$$

For example, in Fig. 8 (a), info maps $\phi^{(1)}$ and $\phi^{(3)}$ are similar to each other and $\mathcal{E}^{(1,3)}$ is small, while $\phi^{(1)}$ and $\phi^{(2)}$ are different from each other and $\mathcal{E}^{(1,2)}$ is large.

Then, an *affine transformed weight space* $\mathcal{B}'$ is constructed as follows. Let $\mathcal{B}'$ be an $(m-1)$-simplex where (i) each corner point of $\mathcal{B}'$ corresponds to an info map $\phi^{(i)}$, and (ii) the line segment connecting two corner points (corresponding to $\phi^{(i)}$ and $\phi^{(j)}$) has length $\mathcal{E}^{(i,j)}$. $\mathcal{B}'$ exists as the ergodic metric in (6), which defines the length of each edge of the simplex $\mathcal{B}'$, is a Sobolev metric [10] and satisfies the triangle inequality.[1]

After specifying a coordinate system to both $\mathcal{B}'$ and $\mathcal{B}$, an affine transformation $\mathcal{A} : \mathcal{B}' \to \mathcal{B}$ can be found by associating each pair of corner points $(p, p'), p \in \mathcal{B}, p' \in \mathcal{B}'$. For each $p' \in \mathcal{B}'$, a corresponding point $\mathcal{A}(p') \in \mathcal{B}$ can be found, and the corresponding weight vector $\vec{w}$ can be obtained based on the coordinate of $\mathcal{A}(p')$. Let $\mathcal{A}_{\vec{w}}(p'), p' \in \mathcal{B}'$ denote the map from the coordinate of a point $p' \in \mathcal{B}'$ to the actual weight vector $\vec{w}$ that will be used to scalarize the info maps, and let $\mathcal{A}_{\vec{w}}^{-1}(\vec{w}), \vec{w} \in \mathcal{B}$ denote the inverse map from a weight vector $\vec{w}$ to the coordinate of a point in $\mathcal{B}'$.

An example when $m = 3$ is shown in Fig. 3 (b). $\mathcal{B}'$ is a triangle, where the three enclosing line segments have lengths $\mathcal{E}^{(1,2)}, \mathcal{E}^{(2,3)}, \mathcal{E}^{(3,1)}$ respectively. A possible coordinate system for $\mathcal{B}'$ is to place the origin at point $x' \in \mathcal{B}'$, align the x-axis with line segment $x'y'$. Then point y' has coordinate $(\mathcal{E}^{(1,2)}, 0)$ and the coordinate of point z' can be determined since the

[1]First, in practice, since K in (6) is often selected as a finite number, the Fourier coefficients ϕ_k of any information map ϕ that appears in (6) must be zero for all $k > K$, so that (6) remains a metric and the triangle inequality holds. Second, it is possible that $\mathcal{B}'$ degenerates and is of dimension less than $(m-1)$. (For example, when $m = 3$, the $(m-1)$-simplex is a triangle. If the three corner points of the triangle are co-linear, the triangle degenerates into a line segment.) For a degenerate case, the proposed adaptive sampling is not applicable while the basic sampling in Sec. III-D still works. For the rest of the presentation, we consider the case where the constructed $(m-1)$-simplex is non-degenerate.

Algorithm 2 Pseudocode for *AdaptiveNeighbor*($\vec{w}$)

1: $O \leftarrow \emptyset$ ▷ The output, a set of weight vectors.
2: $p' \leftarrow \mathcal{A}_{\vec{w}}^{-1}(\vec{w})$
3: $\Delta \leftarrow \{(0, d'), (0, -d'), (d', 0), (-d', 0)\}$ ▷ $m = 3$
4: **for all** $\delta \in \Delta$ **do**
5: $p'_{new} \leftarrow p' + \delta$
6: **if** $p'_{new} \notin \mathcal{B}'$ **then**
7: **continue**
8: Add $\mathcal{A}_{\vec{w}}(p'_{new})$ to O
9: **return** O

length of $x'z'$ and $y'z'$ are both known. With equations $x = \mathcal{A}(x'), y = \mathcal{A}(y'), z = \mathcal{A}(z')$, the affine map $\mathcal{A}$ can be determined.

With $\mathcal{B}'$ and the map $\mathcal{A}_{\vec{w}}$, SL-ES can sample points from $\mathcal{B}'$ (instead of directly sampling weight vectors from $\mathcal{B}$), and each sampled point $p' \in \mathcal{B}'$ can be transformed into a (valid) weight vector $\vec{w} \in \mathcal{B}$, which is then used to compute a scalarized info map. Specifically, as shown in Alg. 2, let O denote the set of sampled weight vectors, which is initialized as an empty set, and let Δ denote the set of possible differences between two neighboring points in $\mathcal{B}'$ using the aforementioned deterministic sampling strategy. Here, we use d' to denote the step size to note the difference from the d in the previous section. Note that, Line 3 in Alg. 2 only shows the Δ when $m = 3$. For each $\delta \in \Delta$, a neighbor point $p'_{new} \leftarrow p' + \delta$ is generated. If p' is still within $\mathcal{B}'$, the corresponding weight vector $\mathcal{A}_{\vec{w}}(p'_{new})$ is added to O. Finally, set O is returned, which contains all sampled neighbor weight vectors. Here, $\mathcal{B}'$ is constructed by considering the difference between info maps, and sampling from $\mathcal{B}'$ allows SL-ES to *adapt* the sampling density to the difference between info maps, which is verified in Sec. V.

F. Discussion

1) Earlier Termination: In practice, when a strong prior preference between info maps (represented by $\vec{w}_{init}$) is available, the termination condition of SL-ES can be modified so that SL-ES terminates earlier when a certain neighborhood of $\vec{w}_{init}$ has been explored. Note that SL-ES explores the weight space $\mathcal{B}$ in a breadth-first manner, and thus SL-ES explores the neighborhood around $\vec{w}_{init}$ in $\mathcal{B}$ at first. This allows SL-ES to quickly compute a set of Pareto-optimal solutions that is "centered" on the prior preference of the user.

2) Weight Space Coverage: SL-ES can be regarded as a framework that converts an MO-ES problem into a "weight space coverage problem": SL-ES iteratively samples weight vectors from $\mathcal{B}$ and terminates when the entire weight space $\mathcal{B}$ is covered. The adaptive sampling method transforms $\mathcal{B}$ into $\mathcal{B}'$ based on the ergodic metrics between info maps and then covers $\mathcal{B}'$. Note that other types of transformation (e.g. non-linear transformation) or different sampling methods can also be leveraged based on the domain knowledge of the application within the proposed SL-ES framework.

G. Extension to Multiple Agents

Let $j = 1, 2, \cdots, N$ denote a set of N homogeneous agents, let q_j denote a trajectory of agent j in the workspace. For the purpose of presentation, within this section, let $q := \{q_1, q_2, \cdots, q_N\}$ denote a joint trajectory of all agents, where each $q_j \in q$ has the same time horizon. Similarly to [10], [22], the Fourier coefficients of the time-averaged statistics of a trajectory defined in (1) can be extended to multiple agents by taking the average over all agents:

$$c_k(x, q) = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{T} \int_0^T F_k(q_j(\tau)) d\tau. \tag{7}$$

Then, the extension of the ergodic metric to multiple agents remains the same as in (2), where c_k is now computed based on (7):

$$\begin{aligned} \mathcal{E}(\phi, q) &= \sum_{k=0}^{K} \lambda_k (c_k - \phi_k)^2 \\ &= \sum_{k=0}^{K} \lambda_k \left(\frac{1}{N} \sum_{j=1}^{N} \frac{1}{T} \int_0^T F_k(q_j(\tau)) d\tau - \phi_k \right)^2 \end{aligned} \tag{8}$$

Finally, the underlying single-objective ergodic search algorithm used by SL-ES and A-SL-ES can be replaced by a multi-agent algorithm, such as [10], [22], in order to address collision avoidance or communication limitation constraints among the agents if needed. The rest part of our framework remains the same. For a multi-agent system, the computed solution (i.e., a joint trajectory) is still guaranteed to be Pareto-optimal (as analyzed in Sec. IV) as long as the underlying multi-agent ergodic search algorithm can minimize (8) with respect to a scalarized info map subject to the agent-agent constraints.

This extension to multiple agents does not consider heterogeneity among agents where each agent has diverse capability to search different info maps, which is beyond the scope of this work.

IV. Analysis

This section proves that the trajectories computed by SL-ES and A-SL-ES are guaranteed to be Pareto-optimal to the MO-ES problem. We begin with a review of the existing scalarization techniques in the MOO literature [17] which is leveraged here, and then present our proof.

A. Mathematical Preliminaries

Given a weight vector $\vec{w} \in \mathcal{B}$, let $\mathcal{E}_w$ denote the weighted-sum of the ergodic vector $\vec{\mathcal{E}}$, which is defined as follows:

$$\mathcal{E}_w(q) := \sum_{i=1}^{m} w^{(i)} \mathcal{E}(\phi^{(i)}, q). \tag{9}$$

Note that $\mathcal{E}_w(q)$ is the weighted-sum of the ergodic metrics $\mathcal{E}(\phi^{(i)}, q), i = 1, 2, \cdots, m$, while the $\mathcal{E}(\phi', q)$ defined in (5) is the ergodic metric with respect to the scalarized map ϕ'.

Proposition 1. *For a weight vector $\vec{w} \in \mathcal{B}$ with $w^{(i)} > 0, \forall i = 1, 2, \cdots, m$, let q^* be a trajectory that minimizes $\mathcal{E}_w(q)$, then q^* is a Pareto-optimal solution, i.e., the corresponding ergodic vector $\vec{\mathcal{E}}(q^*)$ belongs to the Pareto-optimal front.*

This proposition is borrowed from Chapter 3 in [17], where a detailed proof can be found. We present the main idea here to make the paper self-contained. We prove by contradiction. Assume that $\vec{\mathcal{E}}(q^*)$ is not Pareto-optimal. Then, there exists another trajectory q' such that $\vec{\mathcal{E}}(q') \succeq \vec{\mathcal{E}}(q^*)$. Thus, $\mathcal{E}_w(q') = \sum_{i=1}^m w^{(i)}\mathcal{E}(\phi^{(i)}, q') < \sum_{i=1}^m w^{(i)}\mathcal{E}(\phi^{(i)}, q^*) = \mathcal{E}_w(q^*)$. The above inequality holds due to the definition of dominance (Def. 1) and that $w^{(i)} > 0, \forall i = 1, 2, \cdots, m$. It means that q^* does not minimize function $\mathcal{E}_w(q)$, which leads to contradiction.

B. Pareto-Optimality

Given a trajectory q, let $c_k(q)$ denote the Fourier coefficients of q as described in Sec. II. The idea of the proof is to first calculate the difference $\mathcal{E}_w(q) - \mathcal{E}(\phi', q)$, which is independent of q. Therefore, minimizing $\mathcal{E}_w(q)$ is equivalent to minimizing $\mathcal{E}(\phi', q)$, i.e., when $\mathcal{E}(\phi', q)$ reaches the minimum for a certain q, $\mathcal{E}_w(q)$ also reaches the minimum. Then, since the trajectories computed by SL-ES and A-SL-ES minimize $\mathcal{E}(\phi', q)$, these trajectories minimize $\mathcal{E}_w(q)$ and are Pareto-optimal according to Proposition 1.

Proposition 2. *Let q_* denote a trajectory that minimizes $\mathcal{E}(\phi', q)$, then q_* also minimizes $\mathcal{E}_w(q)$.*

Proof. The difference $\mathcal{E}_w(q) - \mathcal{E}(\phi', q)$ is given by

$$
\begin{aligned}
&\mathcal{E}_w(q) - \mathcal{E}(\phi', q) \\
=&\sum_{i=1}^{m}\sum_{k=1}^{K} w^{(i)}\lambda_k \left(c_k(q) - \phi_k^{(i)}\right)^2 \\
&- \sum_{k=0}^{K}\lambda_k \left(c_k(q) - \sum_{i=1}^{m} w^{(i)}\phi_k^{(i)}\right)^2 \\
=&\sum_{i=1}^{m}\sum_{k=1}^{K} w^{(i)}\lambda_k \left(c_k(q)^2 - 2c_k(q)\phi_k^{(i)} + {\phi_k^{(i)}}^2\right) \\
&- \sum_{k=0}^{K}\lambda_k \left(c_k(q)^2 - 2c_k(q)\sum_{i=1}^{m} w^{(i)}\phi_k^{(i)} + \left(\sum_{i=1}^{m} w^{(i)}\phi_k^{(i)}\right)^2\right) \\
=&\sum_{k=1}^{K}\lambda_k \left(c_k(q)^2 \sum_{i=1}^{m} w^{(i)} - 2c_k(q)\sum_{i=1}^{m} w^{(i)}\phi_k^{(i)} + \sum_{i=1}^{m} w^{(i)}{\phi_k^{(i)}}^2\right) \\
&- \sum_{k=0}^{K}\lambda_k \left(c_k(q)^2 - 2c_k(q)\sum_{i=1}^{m} w^{(i)}\phi_k^{(i)} + \left(\sum_{i=1}^{m} w^{(i)}\phi_k^{(i)}\right)^2\right) \\
=&\sum_{k=0}^{K}\lambda_k \left(\sum_{i=1}^{m} w^{(i)}{\phi_k^{(i)}}^2 - \left(\sum_{i=1}^{m} w^{(i)}\phi_k^{(i)}\right)^2\right),
\end{aligned}
$$

which is independent of q. Therefore, if q_* is a trajectory that minimizes $\mathcal{E}(\phi', q)$, then q_* also minimizes $\mathcal{E}_w(q)$. □

Theorem 1. *For a weight vector $\vec{w} \in \mathcal{B}$ with $w^{(i)} > 0, \forall i = 1, 2, \cdots, m$, the trajectory computed by SL-ES and A-SL-ES is Pareto-optimal.*

Proof. With Proposition 1 and 2, if a trajectory q minimizes $\mathcal{E}(\phi', q)$, then q is Pareto-optimal to the MO-ES problem. By design, both SL-ES and A-SL-ES leverage ergodic search algorithms (Line 8 in Alg. 1) to minimize $\mathcal{E}_w(q)$ and the resulting trajectory is thus Pareto-optimal.

□

Remark 2. *Note that the scalarization method in [17] as well as the above proof requires that all components in the weight vector are positive (i.e., non-zero). Intuitively, when a component $w^{(i)}$ is zero, the scalarization method (including our SL-ES) would ignore the i-th objective and only optimize other objectives. Consequently, the computed solution may still be improved w.r.t. the i-th objective without deteriorating any other objective, and the solution is thus not guaranteed to be Pareto-optimal.*

C. Discussion on Completeness

In this paper, an algorithm is called a "complete" algorithm if it can find the entire Pareto-optimal front. It remains an open question whether SL-ES is complete (i.e., whether the entire Pareto-optimal front can be obtained by varying the weight vectors sampled from $\mathcal{B}$). Let $\mathcal{E}_{all}$ denote the set of ergodic vectors corresponding to all feasible trajectories. To show that SL-ES is complete, we need to show that $\mathcal{E}_{all}$ is convex [17].[2] However, the Fourier coefficients of objective functions and the dynamics of the robot under consideration offer additional challenges in determining the convexity of $\mathcal{E}_{all}$.

V. Experimental Results

A. Baseline Methods and Implementation

MOGAs [14], [15] are popular approaches to solve MOO problems, which are also applicable to the MO-ES problem. We use NSGA-II [14], a popular MOGA for MOO problems, as the first baseline approach.[3] A second baseline approach is a naive scalarization method, which differs from SL-ES as it leverages neither the idea of sequential local optimization nor adaptive weight sampling. It iteratively samples $\vec{w} \in \mathcal{B}$, and plans ergodic trajectory w.r.t. the scalarized info map by optimizing from some common naive initial guess, such as a zero control input.

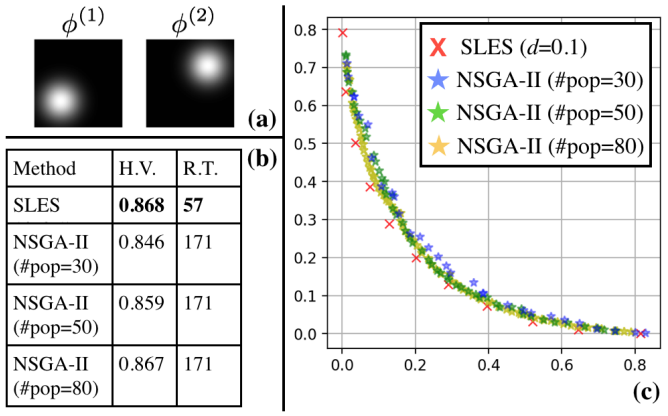
We implement our algorithms[4] and the naive scalarization method in Python, and use the NSGA-II implementation from pymoo [30], a MOGA library, for our experiments. We run tests on a laptop with an Intel Core i7 CPU and 16 GB RAM. All tests have a workspace of size $[0, 1] \times [0, 1]$. We specify the robot dynamics as a differential-drive robot that initially locates at the center of the workspace $(0.5, 0.5)$ with orientation zero (pointing to the right). Specifically, the robot state is $s = (p_x, p_y, \theta)$ and the dynamics is $\dot{s} = (v\cos(\theta), v\sin(\theta), \omega)$ where (v, ω) is the control input vector and represents linear and angular velocities. In our test, the linear velocity is required to be positive at all times.

For presentation purposes, we use "Scala." to denote the naive scalarization method, "SL-ES" to denote our algorithm with the basic neighbor sampling method (Sec. III-D), and "A-SL-ES" to denote our algorithm with the adaptive neighbor

[2]Specifically, we need to analyze the convexity of the set $\{\vec{a} + \vec{b} \mid \forall \vec{a} \in \mathcal{E}_{all}, \forall \vec{b} \geq 0, \vec{b} \in \mathbb{R}^M\}$. We refer the reader to Chapter 3 of [17] for more details.

[3]NSGA-II is popular for MOO problems with two or three objectives. When there are more than three objectives (sometimes referred to as "many-objective optimization"), NSGA-III can be used.

[4]Our code is available at https://github.com/wonderren/public_moes



Method	H.V.	R.T.
SLES	**0.868**	**57**
NSGA-II (#pop=30)	0.846	171
NSGA-II (#pop=50)	0.859	171
NSGA-II (#pop=80)	0.867	171

Fig. 4. (a) shows the two info maps to be covered. (b) shows the hyper-volume (H.V.) of the solution set computed by each method, where we allow NSGA-II (baseline) to run for *three* times the run time (R.T., in seconds) of SL-ES. (c) visualizes the ergodic vectors of the computed solutions. SL-ES computes a set of solutions with similar or better quality than NSGA-II while using only one third of the run time of NSGA-II.

sampling method (Sec. III-E). We set a termination threshold $\epsilon = 10^{-3}$ for each *ErgodicSearch* call in Alg. 1: when the ergodic metric w.r.t. the scalarized info map is no larger than ϵ, *ErgodicSearch* terminates. To describe the quality of the computed Pareto-optimal front, we use the "hyper-volume" indicator (H.V.) [15] from the MOO literature. Intuitively, H.V. denotes the volume enclosed by the computed Pareto-optimal front and a reference point in the objective space, which is set to $(1, 1, \cdots, 1)$ for all tests.

B. Comparison with NSGA-II

We begin our tests with $m = 2$, and the info maps are shown in Fig. 4(a). We compare SL-ES with NSGA-II. We measure the run time of SL-ES (denoted as T_1) and let NSGA-II run for *three* times the run time of SL-ES (i.e., $3T_1$). As shown in Fig. 4, increasing the "population size" (a hyper-parameter in NSGA-II) can slightly improve the solution quality. However, SL-ES computes a set of solutions with similar or better quality (in terms of H.V.) than NSGA-II while using only one third of the run time of NSGA-II. The possible reason is, while being general to various problems, NSGA-II treats the objective functions as a "black-box" and often ignores the underlying structure of the problem (such as the dynamics of the robot and the local metric structures).

C. Comparison with Naive Scalarization

We then compare SL-ES against the naive scalarization method (Scala.) with the same test settings as in the previous section. In Fig. 5, the horizontal axis indicates the number of optimization iterations in the *ErgodicSearch* procedure while the vertical axis denotes the ergodic metric in (5). Note that at the beginning of each episode, a different $\vec{w}$ (and thus a different ϕ') is considered, and thus the ergodic metric changes. As shown in Fig. 5, Scala. takes the most number of optimization iterations in each episode since it always starts from the same naive initial guess (i.e., a zero control input). Both SL-ES and A-SL-ES run faster than Scala. especially when d decreases (which means there are more planning episodes). Take Fig. 5(b) for example, SL-ES requires less than half of the run time in comparison with Scala., and still computes a solution set with the same quality in terms of H.V. It shows that running local optimization by (i) sampling weight vectors that are near to each other, and (ii) reusing the solution from the previous episodes as the initial guess for the current episode, can expedite the computation.

Fig. 5 also demonstrates the benefit of the proposed adaptive neighbor sampling: specifying d in $\mathcal{B}$ is not intuitive and can lead to either too sparse ($d = 0.2$) or too dense sampling ($d = 0.05$), which leads to either a low H.V. value or a large number of episodes. Sampling based on d' in the affine transformed weight space $\mathcal{B}'$ allows the algorithm to adapt to the differences between info maps. Additionally, d' has the same unit as the ergodic metric between info maps, and is thus more intuitive to specify.

D. Different Sampling Step Sizes

This section tests A-SL-ES with varying step sizes d', with $m = 2$, and with the same info maps as in the previous section. As shown in Fig. 6, by tuning d', there is a trade-off between H.V. values, which indicate the quality of the solution set, and the computational burden, which is indicated by the run time. Having slightly larger d' can speed up the computation significantly with small decrease in H.V.

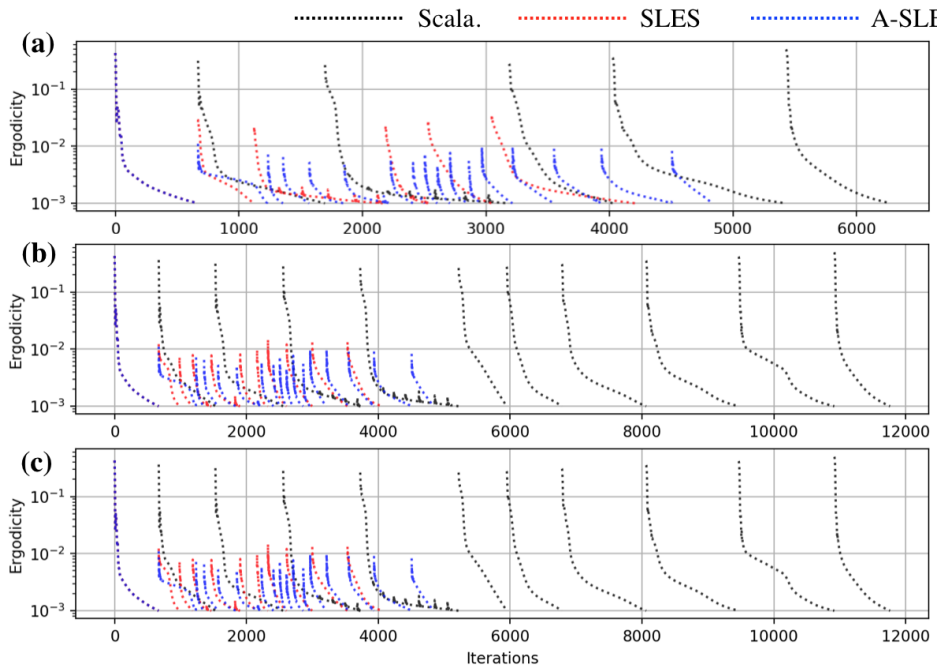
E. Various Info Maps

We further test the algorithms using various info maps as shown in Fig. 7, where each info map is a mixture of (two-dimensional) Gaussian distributions with randomly sampled expectations and covariance matrices. Among these methods, Scala. and SL-ES are tested with $d = 0.1$, NSGA-II has a fixed population size of 30, and A-SL-ES has $d' = 0.05$. We observe from Fig. 7 that, our approach SL-ES computes solutions with better hyper-volume than the NSGA-II baseline within the same amount of runtime. Additionally, our SL-ES and A-SL-ES compute solutions with similar quality as the Scala. baseline, while running up to an order of magnitude faster than Scala.

F. Three Objectives

We then test NSGA-II, SL-ES and A-SL-ES with $m = 3$. The info maps are shown in Fig. 8(a). Note that $\phi^{(1)}$ is similar to $\phi^{(3)}$ while they are both quite different from $\phi^{(2)}$. Fig. 8(d) shows that A-SL-ES provides a subset of the Pareto-optimal front of similar quality in comparison with the results computed by SL-ES (in terms of H.V.) while having a much smaller run time. From Fig. 8(b) and 8(c), it is obvious that A-SL-ES can adaptively sample weight vectors based on the difference between each pair of info maps: there are only a few blue points in Fig. 8(c) to represent the Pareto-optimal front. In contrast, SL-ES has a lot of samples (the red points in Fig. 8(b)) to represent the Pareto-optimal front.

We note that the current implementation of the algorithm can take relatively long runtime to compute a set of Pareto-optimal solutions (e.g. Fig. 8(d)), which is caused by (i) the

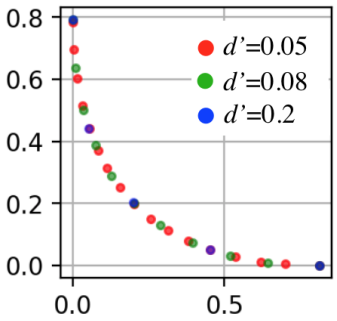


Method	H.V.
Scala. (d=0.2)	0.838
SLES (d=0.2)	0.839
A-SLES (d'=0.05)	0.877

Method	H.V.
Scala. (d=0.1)	0.867
SLES (d=0.1)	0.867
A-SLES (d'=0.05)	0.877

Method	H.V.
Scala. (d=0.05)	0.880
SLES (d=0.05)	0.880
A-SLES (d'=0.05)	0.877

Fig. 5. The horizontal axis indicates the number of optimization iterations in the *ErgodicSearch* procedure while the vertical axis denotes the ergodic metric in (5). Note that at the beginning of each episode, a different $\vec{w}$ (and thus a different ϕ') is considered, and thus the ergodic metric "jumps". The corresponding tables in (a), (b) and (c) show the hyper-volumes of different methods with different step sizes. This figure shows that SL-ES requires obviously less computational time than Scala. (i.e., baseline) to compute a set of solutions with similar quality in terms of H.V. More discussion can be found in the text.



A-SLES	H.V.	R.T.
d'=0.05	**0.878**	34
d'=0.08	0.870	21
d'=0.2	0.840	**13**

Fig. 6. Hyper-volume (H.V.) and run time (R.T.) (in seconds) of A-SL-ES with varying sampling step size d'. This figure shows, by tuning d', A-SL-ES can trade-off between solution quality and run time. Having slightly larger d' can speed up the computation with small decrease in the H.V.

difficulty of the MO-ES problem, and (ii) the implementation issue. Specifically, first, solving a multi-objective optimization problem is in general computationally expensive since a set of Pareto-optimal solutions (rather than a single solution) are desired. Second, our implementation is not optimized for running speed. We point out possible techniques that can expedite the computation in Sec. VI.

G. Robot Simulation

We apply the proposed A-SL-ES algorithm to an example MO-ES problem and simulate the computed trajectory in ROS.[5] The example involves a warehouse with hazardous gas leakage. The goal is to find both sources of leakage and search for survivors. The two objectives are described using two info maps, which can be generated based on the prior knowledge of the warehouse (Fig. 9). Usually these two objectives cannot be optimized simultaneously as survivors can be far away from the gas leakage source. We use A-SL-ES to compute a set of Pareto-optimal trajectories, which can then be visualized to the decision maker on site so that a more informed decision can be made. For example, if the effect of the gas for humans is minor but it affects the goods in the warehouse significantly, one might want to choose a trajectory that prioritizes finding the leakage source more than searching for humans inside.

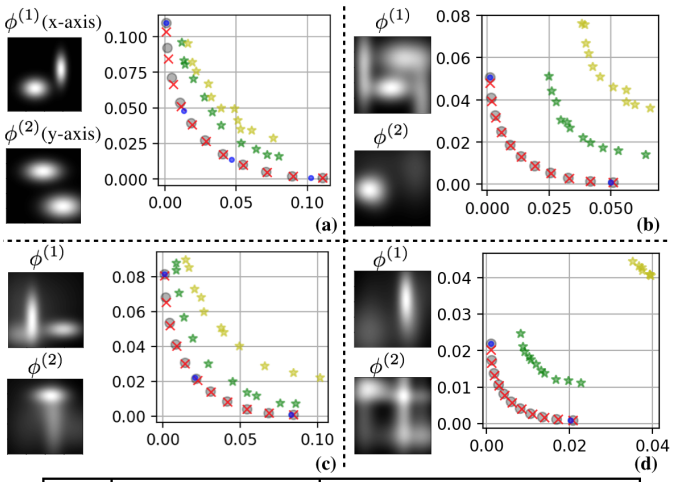
Fig. 9(b) visualizes three Pareto-optimal solutions. For instance, the green trajectory prioritizes finding survivors (the pink info map) while the red one favors localizing leakage sources (the yellow info map). Please refer to our multi-media attachment for more details.

H. Test with Two Physical Robots

To verify that the planned trajectories can be executed on physical robots, we run tests with two ROSBots (Fig. 10(c)), a different-drive wheeled robot with the ROS navigation stack installed.[6] ROSBot is equipped with a 2D Lidar for localization, and we first build a 2D occupancy grid map by manually commanding a ROSBot to move around the

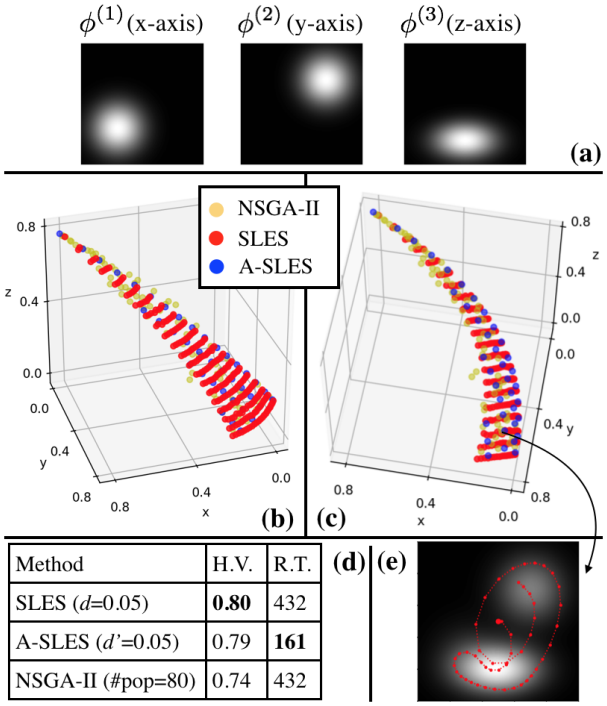
[5]Our ROS implementation leverages https://github.com/wh200720041/warehouse_simulation_toolkit and https://github.com/bostoncleek/ergodic_exploration.

[6]https://husarion.com/manuals/rosbot/



	Algorithms	(a)	(b)	(c)	(d)
H.V.	Scala.	0.996	0.997	0.997	0.998
	SLES (ours)	0.996	0.997	0.997	0.998
	A-SLES (ours)	0.994	0.996	0.995	0.998
	NSGA-II (R.T. Scala.)	0.970	0.961	0.983	0.981
	NSGA-II (R.T. SLES)	0.954	0.926	0.961	0.926
R.T. Ratio	SLES / Scala.	25.8%	15.9%	16.2%	17.6%
	A-SLES / Scala.	12.4%	10.9%	13.5%	8.3%

Fig. 7. Comparison of methods with various info maps. Figure (a,b,c,d) show the approximated Pareto-optimal front computed by various methods. Among them, the runtime of Scala. and SL-ES are recorded and denoted as R.T. Scala. and R.T. SL-ES respectively. NSGA-II is then given a runtime budget of R.T. Scala. (green stars) and R.T. SL-ES (yellow stars) respectively. The table shows both the hyper-volume (H.V.) and the runtime (R.T.) ratios. Our methods (SL-ES and A-SL-ES) compute better quality solution than the NSGA-II baseline, while running up to an order of magnitude faster than the Scala. baseline.



Method	H.V.	R.T.
SLES (d=0.05)	**0.80**	432
A-SLES (d'=0.05)	0.79	**161**
NSGA-II (#pop=80)	0.74	432

Fig. 8. (a) shows the three info maps to be covered. (b) highlights the solution (red) computed by SL-ES and (c) highlights the solution (blue) computed by A-SL-ES. (d) shows the hyper-volume and run time (in seconds) of each method. (e) shows a scalarized info map and the corresponding ergodic trajectory. A-SL-ES computes solutions of similar quality while requiring less than half of the run time in comparison with SL-ES and NSGA-II.

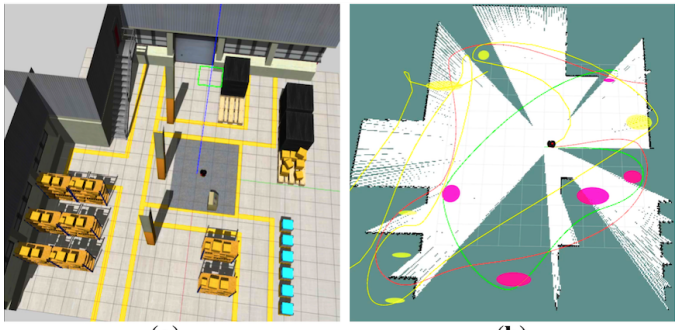


Fig. 9. (a) shows the warehouse environment and (b) shows the information maps visualized as the yellow (probable gas leakage locations) and pink (probably survivor locations) markers on RViz. The current method does not consider obstacle avoidance during the ergodic planning and our simulation relies on an additional local planner to avoid obstacles.

workspace and then copy the map to both the robots for localization. As shown in Fig. 10(a), we consider two info maps and run SL-ES to cover them. We then randomly pick a solution, which corresponds to the weight vector $(0.8, 0.2)$, for execution. To execute the planned trajectories using the ROS navigation stack available on ROSBot, we down-sample the trajectory and send the resulting waypoints to ROSBot to follow, rather than sending velocity or acceleration commands to the robots. As a result, the robot may slow down as they are close to a waypoint before moving to the next waypoint. Please refer to our multi-media attachment for a visualization.

VI. Conclusion and Future Work

This paper formulates a Multi-Objective Ergodic Search (MO-ES) problem, which requires planning trajectories to simultaneously cover multiple info maps. To solve the MO-ES problem, we propose a framework called Sequential Local Ergodic Search (SL-ES). SL-ES scalarizes info maps rather than the objective functions using a weight vector, which allows us to leverage the existing various (single-objective) ergodic search algorithms to plan the trajectory. To obtain weight vectors, SL-ES explores the weight space (the space that contains all possible weight vectors) in a breadth-first manner and leverages the idea of local optimization by (i) sampling new weight vectors in the neighborhood of the current weight vector, and (ii) optimize the trajectory corresponding to the new weight vector by using the current solution as the initial guess. Additionally, to further expedite SL-ES, we also develop a variant called Adaptive SL-ES (A-SL-ES) that can adjust the density of sampled weight vectors based on a similarity metric between each pair of info maps to be covered in the Fourier coefficient space. We prove that the solutions computed by SL-ES and A-SL-ES are guaranteed to be Pareto-optimal. The numerical results verify the advantages of SL-ES and A-SL-ES over the baselines. The simulation and the physical robot tests verify that the planned results can be executed on real robots.

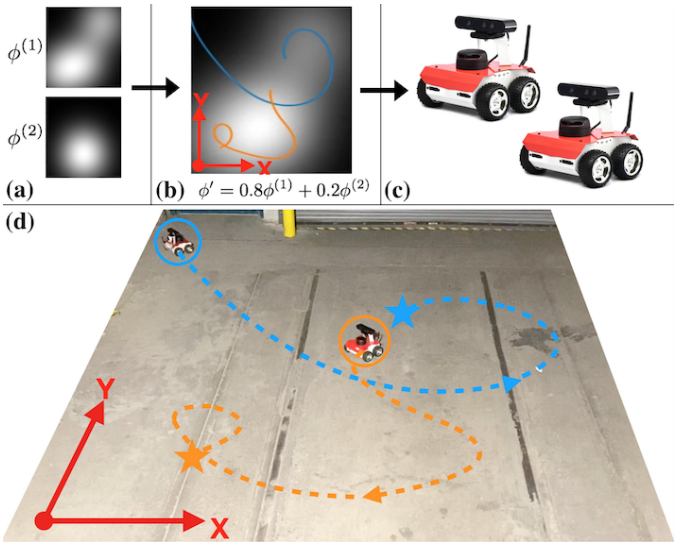


Fig. 10. (a) shows the two info maps to be covered. (b) shows the scalarized info map and the planned trajectories for both agents. (c) shows the two ROSBot used in the test. (d) shows the test site and visualize the trajectories to be followed by both robots, where the stars mark the ending positions of the robots.

Future Work. This paper is a first attempt to investigate MO-ES problems, and considers multiple static info maps without obstacles. It is worthwhile to investigate variants of the MO-ES problem where the info maps are dynamic (i.e., maps are updated in an online manner during the robot motion) or the workspace is cluttered with static and dynamic obstacles. Additionally, one can also consider heterogeneous multi-agent systems where each agent has diverse capability to search the info maps. Finally, to expedite the computation, one can consider using C++ rather than Python for implementation, and leveraging real-time ergodic search techniques [23], especially for time critical tasks such as search and rescue.

ACKNOWLEDGMENTS

We sincerely thank Dr. Geordan Gutow in the Biorobotics Lab at Carnegie Mellon University for his valuable inputs on this paper.

REFERENCES

[1] K. Lee, S. Martínez, J. Cortés, R. H. Chen, and M. B. Milam, "Receding-horizon multi-objective optimization for disaster response," in *2018 Annual American Control Conference (ACC)*. IEEE, 2018, pp. 5304–5309.

[2] 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.

[3] W. Chen and L. Liu, "Pareto monte carlo tree search for multi-objective informative planning," in *Proceedings of Robotics: Science and Systems*, FreiburgimBreisgau, Germany, June 2019.

[4] 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.

[5] 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.

[6] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull, "Morse decompositions for coverage tasks," *The international journal of robotics research*, vol. 21, no. 4, pp. 331–344, 2002.

[7] M. Santos, Y. Diaz-Mercado, and M. Egerstedt, "Coverage control for multirobot teams with heterogeneous sensing capabilities," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 919–925, 2018.

[8] 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.

[9] 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.

[10] 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.

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

[12] 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, 2016.

[13] J. Casper and R. Murphy, "Human-robot interactions during the robot-assisted urban search and rescue response at the world trade center," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 3, pp. 367–385, 2003.

[14] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[15] M. T. Emmerich and A. H. Deutz, "A tutorial on multiobjective optimization: fundamentals and evolutionary methods," *Natural computing*, vol. 17, no. 3, pp. 585–609, 2018.

[16] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, no. 6, pp. 712–731, 2007.

[17] M. Ehrgott, *Multicriteria optimization*. Springer Science & Business Media, 2005, vol. 491.

[18] Z. Ren, A. Srinivasan, H. Coffin, I. Abraham, and H. Choset, "A Local Optimization Framework for Multi-Objective Ergodic Search," in *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022.

[19] E. Ayvali, H. Salman, and H. Choset, "Ergodic coverage in constrained environments using stochastic trajectory optimization," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 5204–5210.

[20] I. Abraham, A. Prabhakar, and T. D. Murphey, "An ergodic measure for active learning from equilibrium," *IEEE Transactions on Automation Science and Engineering*, 2021.

[21] A. Kalinowska, A. Prabhakar, K. Fitzsimons, and T. Murphey, "Ergodic imitation: Learning from what to do and what not to do," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3648–3654.

[22] I. Abraham and T. D. Murphey, "Decentralized ergodic control: distribution-driven sensing and exploration for multiagent systems," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2987–2994, 2018.

[23] A. Mavrommati, E. Tzorakoleftherakis, I. Abraham, and T. D. Murphey, "Real-time area coverage and target localization using receding-horizon ergodic exploration," *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 62–80, 2017.

[24] Z. Ren, S. Rathinam, M. Likhachev, and H. Choset, "Multi-objective safe-interval path planning with dynamic obstacles," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8154–8161, 2022.

[25] Z. Ren, R. Zhan, S. Rathinam, M. Likhachev, and H. Choset, "Enhanced multi-objective A* using balanced binary search trees," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 15, no. 1, 2022, pp. 162–170.

[26] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[27] L. Nardi, D. Koeplinger, and K. Olukotun, "Practical design space exploration," in *2019 IEEE 27th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 2019, pp. 347–358.

[28] Z. Ren, S. Rathinam, and H. Choset, "A conflict-based search framework for multiobjective multiagent path finding," *IEEE Transactions on Automation Science and Engineering*, pp. 1–13, 2022.

[29] H. Salman, E. Ayvali, and H. Choset, "Multi-agent ergodic coverage with obstacle avoidance," in *Twenty-Seventh International Conference on Automated Planning and Scheduling*, 2017.
[30] J. Blank and K. Deb, "pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

Zhongqiang (Richard) Ren (Student Member, IEEE) received the Ph.D. and M.S. degree from Carnegie Mellon University, Pittsburgh, PA, USA, and the dual B.E. degree from Tongji University, Shanghai, China, and FH Aachen University of Applied Sciences, Aachen, Germany. He is currently a Postdoctoral Fellow in the Biorobotics Lab, Robotics Institute, Carnegie Mellon University. His research focuses on solving fundamental path and motion planning problems for single and multiple robots and advancing the frontier of multi-agent planning, heuristic search and optimal control.

Akshaya Kesarimangalam Srinivasan received a bachelor's degree in Computer Science and Engineering from the National Institute of Technology, Tiruchirappalli, India, in 2020. She is pursuing a Robotics master's degree at Carnegie Mellon University, Pittsburgh, PA, USA. She is currently working on multi-agent multi-objective exploration focused on task allocation and optimization, with the Biorobotics lab, Carnegie Mellon University.

Bhaskar Vundurthy (Member, IEEE) received the B.E. degree (Hons.) in electronics and instrumentation engineering and the M.Sc. degree (Hons.) in chemistry from the Birla Institute of Technology and Science (BITS)-Pilani, Pilani, India, and the M.Tech. degree in control and instrumentation and the Ph.D. degree in robotics, electrical engineering from IIT Madras, Chennai, India.

He then joined the MathWorks Inc., where he developed advanced examples for robotic applications in the areas of deep learning and autonomous driving, followed by a postdoctoral fellowship at University of Notre Dame where he developed hybrid strategies for intelligent players in a game theoretic setting. He is currently a project scientist with The Robotics Institute, Carnegie Mellon University, PA, USA, where his research interests include multiagent planning and control, game theory, computational geometry, and robotics.

Ian Abraham received the B.S. degree in mechanical and aerospace engineering from Rutgers University, New Brunswick, NJ, USA, in 2014 and the M.S. and Ph.D. degrees in mechanical engineering from Northwestern University, Evanston, IL, USA, with the Center for Robotics and Biosystems, in 2017, and 2020, respectively. He is an Assistant Professor in Mechanical Engineering at Yale University, New Haven, CT, USA, and leads the Intelligent Autonomy Lab where his research focuses on developing methods at the intersection of optimal control and learning. Previously, he was a postdoctoral scholar in the Biorobotics Lab, Robotics Institute, Carnegie Mellon University.

Howie Choset (Fellow, IEEE) received the undergraduate degrees in computer science and business from the University of Pennsylvania, Philadelphia, PA, USA, and the M.S. and Ph.D. degrees in mechanical engineering from Caltech, Pasadena, CA, USA. He is a Professor in the Robotics Institute, Carnegie Mellon, Pittsburgh, PA, USA.