Multirobot path-aware global optimization

Tudor Sântejudean[a,b,*], Maria Ceapă[a], Radu Herzal[a], Elvin Pop[a], Vineeth Satheeskumar Varma[a,b], Irinel-Constantin Morărescu[a,b], Lucian Bușoniu[a]

[a]*Technical University of Cluj-Napoca, Automation Department, Memorandumului 28, 400114 Cluj-Napoca, Romania (e-mails: {tudor.santejudean, maria.ceapa, elvin.pop, lucian.busoniu}@aut.utcluj.ro, herzal.eu.radu@student.utcluj.ro),*
[b]*Université de Lorraine, CNRS, CRAN, F-54000 Nancy, France. {vineeth.satheeskumar-varma, constantin.morarescu}@univ-lorraine.fr.*

Abstract

We propose Voronoi Simultaneous Optimistic Optimization (VSOO), a divide-the-best-based method for multirobot global optimization of a Lipschitz-continuous physical objective (e.g., quantity of material, density of litter, signal power), whose Lipschitz constant is unknown. In this problem, a team of mobile robots must autonomously navigate as quickly as possible to all global optima of the objective function defined over their operating area. The objective can have multiple local and global optima, is initially unknown, and can only be evaluated online at robot locations. VSOO utilizes Voronoi partitions driven by the samples collected so far by the robots, which allows them to incrementally refine the search space in their simultaneous search for the optima. We guarantee everywhere-dense and global convergence for any function, and analyze convergence rates for some representative classes of function shapes. Extensive numerical simulations, performed on established classes of benchmark test functions, demonstrate that VSOO approaches all global optima faster than a series of representative source/extremum seeking techniques that – similarly to VSOO – are global optimizers designed for mobile robots. In terms of execution time, VSOO is competitive with these baselines. We finally validate VSOO in real-robot experiments in which TurtleBot3 robots successfully search for the strongest antenna signals indoors.

Keywords: Multirobot systems, divide-the-best global optimization, source seeking, convergence analysis

1. Introduction

We consider a problem in which a team of autonomous mobile robots must work together to find as quickly as possible the global maxima of an initially unknown function. The function to be optimized is defined over the physical search space of the robots, is allowed to have multiple local and global maxima, and must be Lipschitz-continuous. The function can in practice be a physical quantity like pollutant concentration, litter density, signal strength, sound level, etc. Robots are equipped with sensors that enable them to take function samples at their current positions, and have access to their absolute positions (e.g. they operate in a GPS-enabled environment). Robot dynamics, such as limited velocity, restrict future samples to a neighborhood of the robots' current positions. To reach closer to the optima, the path of the robots must be adapted online using the samples collected so far. Since the robots typically do not have sufficient computational power onboard, at every iteration a centralized controller collects all function samples, computes the next sampling location for all robots, and sends these positions to the robots for navigation.

We call the setting described above *multirobot path-aware global optimization* (MR-PAO), an extension of the single-robot case previously studied by Sântejudean and Buşoniu (2022); Sântejudean et al. (2025). This setting encompasses multiple practical applications, such as robots localizing the densest odor plumes (Bourne et al., 2019; Jing et al., 2021) to discover toxic gas leaks or pollutant sources (Bayat et al., 2016; Fu et al., 2019; Francis et al., 2022), robots searching for the highest acoustic level in search and rescue missions (Basiri et al., 2012; Hoshiba et al., 2017; Sibanyoni et al., 2018), or for the strongest received signal to help localizing a transmitter (Graefenstein and Bouzouraa, 2008; Fink and Kumar, 2010) to relay data efficiently over a meshed network of antennas (Nguyen et al., 2003; Saitou et al., 2013). Another application is searching for the largest density of marine litter (e.g. for subsequent collection) with underwater and aerial drones (Vasilijević et al., 2017; Veettil et al., 2022; Šiljeg et al., 2023; Kim, 2023).

Our goals in this work are twofold. First, we aim to develop a novel method that tackles MR-PAO and guarantees convergence to all global optima of the Lipschitz-continuous objective function, without having access to the Lipschitz constant, and in general at faster rates than those given by uniformly sampling the entire search space (Munos, 2011; Sergeyev et al., 2013; Munos, 2014). Here, Lipschitz continuity will be used to guarantee upper bounds on the objective in certain regions, directing the search towards regions with large upper bounds that are more likely to contain optima (Sergeyev, 1998; Munos, 2014). The second goal is to empirically outperform existing source/extremum seeking methods that are applicable to our problem setting, i.e. they optimize the objective while taking into account the dynamics of the robots (Zhang and Ordóñez, 2011; Azuma et al., 2012; Scheinker, 2024).

[*]Corresponding first author.

Literature review

The global Lipschitz optimization of multidimensional, multiextremal, black-box functions has been under intensive research in the last half-century (Strongin, 1978; Jones et al., 1993; Pintér, 1995; Floudas et al., 2005; Stripinis and Paulavičius, 2024), leading to the development of numerous techniques, such as smooth-bounding approaches with exact or estimated Lipschitz constants (Strongin and Sergeyev, 2013; Kvasov and Sergeyev, 2015), methods using space-filling curves (Sergeyev et al., 2013, 2024), simplicial partitioning (Paulavičius and Žilinskas, 2014), optimistic optimization (Munos, 2011, 2014), among many others. In general, these methods iteratively subdivide the search domain into progressively finer regions, selecting for division regions associated with a best characteristic constructed based on function evaluations and region sizes. Such approaches were unified into a general framework called "Divide-the-Best" (Sergeyev, 1998; Kvasov and Sergeyev, 2015; Sergeyev and Kvasov, 2015). Most of these methods can be divided in two large classes: algorithms using exact knowledge of the Lipschitz constant to build bounds on the objective (Strongin and Sergeyev, 2013; Munos, 2014; Paulavičius and Žilinskas, 2014), and methods using global or local estimates of the Lipschitz constant derived from continuous function evaluations (Strongin and Sergeyev, 2013; Sergeyev et al., 2013; Munos, 2014; Kvasov and Sergeyev, 2015). A subtype of the second class are methods that only implicitly estimate the Lipschitz constant by working a priori for any such constant, like in the well-established DIRECT-based algorithms (Paulavičius and Žilinskas, 2014; Sergeyev and Kvasov, 2017; Stripinis and Paulavičius, 2023, 2024). Methods that adapt their Lipschitz estimates locally are expected to work significantly faster compared to methods using exact or global Lipschitz estimates, as they are less prone to overestimating the Lipschitz constant (Sergeyev, 1998; Sergeyev and Kvasov, 2006; Strongin and Sergeyev, 2013). Although global optimizers exist that do not rely on Lipschitz continuity, such as Bayesian optimization (Mockus, 2005; Lizotte, 2008; Zhigljavsky and Žilinskas, 2021) or metaheuristic techniques (Dorigo and Blum, 2005; Boussaïd et al., 2013; Tomar et al., 2024), this work specifically focuses on the class of optimizers that do make this assumption about the objective function.

The global optimization techniques mentioned above have the disadvantage of not being directly applicable to MR-PAO, as they do not take into account the dynamics of the robots and instead sample the space at arbitrary locations. Thus, robots would be assumed to "teleport" to these new states, or – stated differently – samples taken along the way there would be ignored. Our key novelty in this context is a global optimization technique that takes into account the dynamics of the robots (specifically, limited velocity).

Closest to the MR-PAO control-oriented setting is the class of source and extremum seeking methods we already mentioned (Tan et al., 2010; Azuma et al., 2012; Scheinker, 2024). These methods aim to optimize the steady-state process variable of a dynamic system in a model-free fashion, when only online measurements of the objective are available (Tan et al., 2010; Scheinker and Krstić, 2017; Wang et al., 2022). Many techniques have been proposed to achieve this, among which approximate model-based or model-free gradient climbing (Atanasov et al., 2012; Khong et al., 2014a; Poveda and Krstić, 2020), simultaneous-perturbation stochastic approximation (Azuma et al., 2012; Ramirez-Llanos and Martinez, 2018), sliding mode control (Zhu et al., 2014; Zhang et al., 2023), particle swarm (Zou et al., 2015; Gronemeyer et al., 2017) or Bayesian optimization (Bourne et al., 2019; Li et al., 2021; Ghassemi et al., 2022). Different from MR-PAO, when applied to mobile robots, source and extremum seeking usually assume the objective function to be (at least once) differentiable everywhere (Tan et al., 2010; Fu and Özgüner, 2011), possibly radially decaying around the optimum (Atanasov et al., 2012; Zhu et al., 2013), and with a unique global optimum (Azuma et al., 2012; Ghadiri-Modarres and Mojiri, 2020; Li et al., 2021). The last assumption on the objective function is particularly restrictive compared to our setting; indeed, if the function has multiple global optima, only local or semi-global optimization can be achieved (Fu and Özgüner, 2011; Gronemeyer et al., 2020). However, source/extremum seeking considers more complex robot dynamics than we do here, e.g. higher-order models that account for inertia, friction, and other physical effects (Khong et al., 2014a; Zhu et al., 2014; Scheinker, 2024). Often, practical asymptotic stability, which guarantees convergence to a small neighborhood around a global optimum, is sought (Krstić and Wang, 2000; Tan et al., 2010; Khong et al., 2014b; Scheinker, 2024), instead of finding all global optima which is the goal in MR-PAO. Another point of differentiation is that source/extremum seeking do not store past trajectory samples nor do they consider absolute positioning to be available, i.e. robots are not equipped with GPS-like localization systems (Tan et al., 2010; Azuma et al., 2012; Ghadiri-Modarres and Mojiri, 2020). Nevertheless, exceptions exist that aim to find global optima using absolute positioning, like in the DIRECT-based method of Khong et al. (2014b), the Particle Swarm Optimization method of Zou et al. (2015), and the Bayesian Optimization-based method of Ghassemi et al. (2022). These methods will be used as baselines here.

Additional work on global optimization applied to robotics can be found in applications such as determining the working space of robot manipulators by means of space-filling curves (Lera et al., 2021, 2024). Other optimization paradigms designed for mobile robots can be found in coverage/informative path planning, in which mobile robots maximize information gathered about an unknown environment by continuously adapting their route based on collected samples (Schmid et al., 2020; Tan et al., 2021). Another sampling-based approach for autonomous exploration is simultaneous localization and mapping (Grisetti et al., 2010), in which robots equipped with sensors such as LiDARs (Hess et al., 2016; Xu et al., 2022) or depth cameras (Castaneda et al., 2011; Taketomi et al., 2017), try to reconstruct a map of the surrounding environment while simultaneously localizing the robot. Different from MR-PAO, which focuses on locating global maxima, these methods aim to provide a complete map of the environment.

Proposed method and contribution

The key research gap we address here is the lack of an approach for MR-PAO that provably finds all global optima of a Lipschitz-continuous function and provides convergence rates, without sacrificing practical performance. To this end, we adapt a divide-the-best algorithm called Simultaneous Optimistic Optimization (SOO) (Munos, 2011, 2014) to solve MR-PAO. SOO hierarchically partitions the search space so that deeper sets are smaller. If a deeper set has a smaller sample of the objective than a shallower one, it cannot have the largest upper bound no matter the value of the Lipschitz constant: it is dominated by that shallower set. At each iteration, SOO refines all undominated sets. As discussed above for global optimization, SOO and more generally DIRECT-based methods (Jones et al., 1993; Sergeyev and Kvasov, 2017; Stripinis and Paulavičius, 2024) suffer from a key weakness in MR-PAO: they impose an exact partitioning scheme which requires to sample only the cell centers at arbitrarily far apart positions. Therefore, instead of imposing an exact partition shape, we use a Voronoi partition driven by all the samples seen so far, leading to a method which we call Voronoi SOO (VSOO). Similarly to SOO, using cell sizes and sample objective values, VSOO only selects undominated cells for expansion, which for some Lipschitz constant may have the largest upper bounds and therefore the highest chances of containing global optima. Differently from SOO, robots are carefully assigned to the cells selected for refinement, taking into account their limited velocity and making sure that refinement ensures contraction of cell sizes. Robots are divided in two types: exploration robots which focus on wide cells, and exploitation robots which focus on cells with large objective values. Since we deal with physical robots, we mainly study VSOO in dimensions 1-3, although the method may be extended to higher dimensions.

Our analysis shows that VSOO ensures everywhere-dense and global convergence (Sergeyev, 1998), consequently reaching arbitrarily close to all global optima. We further aim for stronger guarantees, and so characterize *convergence rates* to the global optima for two representative classes of function shapes. The core novelty and contribution of this work consists of the VSOO method itself, as well as in its analytical convergence guarantees and rates. In particular, convergence rates are rare and have mostly been proposed for global Lipschitz optimization without control objectives (Sergeyev et al., 2013; Munos, 2014). One exception is our earlier work (Sântejudean et al., 2025), in which rates are given for PAO but only in the single-agent case.

In extensive numerical simulations performed on benchmark optimization functions including GKLS-generated test functions (Gaviano et al., 2003; Stripinis and Paulavičius, 2024), VSOO is shown to outperform a series of representative source/extremum seeking techniques (Khong et al., 2014b; Zou et al., 2015; Ghassemi et al., 2022) by approaching all global optima faster on average, with competitive execution times. We also validate VSOO using our new real-robot experimental testbed, in which a team of TurtleBot3 robots successfully searches for the strongest antenna signal indoors.

Next, Section 2 formally defines the problem and provides more background on SOO. Section 3 presents the VSOO method, while Section 4 presents the analytical convergence guarantees and rates. Then, VSOO is compared against representative source seeking baselines in numerical simulations in Section 5, and validated in real-robot experiments in Section 6. Finally, Section 7 concludes the paper.

2. Problem statement and preliminaries

This section describes the MR-PAO problem, and then briefly introduces the SOO algorithm, which represents the foundation of this work.

Consider first a compact and convex search space $X \subset \mathbb{R}^n$, with $n \in \{1, 2, 3\}$, over which the objective function $f : X \to \mathbb{R}$ is defined. The following assumption on f is imposed.

Assumption 1. ***Lipschitz continuity:*** *The objective function f is globally Lipschitz-continuous:*

$$\|f(x_1) - f(x_2)\| \leq M\|x_1 - x_2\|, \forall x_1, x_2 \in X, \tag{1}$$

where $\|\cdot\|$ represents the 2-norm and $M \in (0, \infty)$ is the Lipschitz constant.

Remark 1. *The global Lipschitz continuity could be relaxed to hold only locally around the global optima, while the 2-norm could be replaced by any semi-metric (Munos, 2014). These settings were chosen in this paper for convenience and ease of presentation.*

Remark 2. *A global optimizer typically requires a certain smoothness of the objective function (Fu and Özgüner, 2011; Suttner and Krstic, 2023), like Lipschitz continuity (Liu and Krstic, 2010; Gronemeyer et al., 2020). Without this assumption, there could be arbitrarily small regions with abrupt changes in function values that might conceal a global optimum. Many real-world phenomena, such as the diffusion of pollutants in the atmosphere or the spread of heat in a medium, exhibit smooth behavior over time, making this assumption reasonable (Gronemeyer et al., 2020).*

Consider now p mobile robots that search for the maxima $x^* \in X^* := \arg\max_{x \in X} f(x)$ inside their operating area, which is equal to X. Each robot $q \in \{1, ..., p\}$ is described in discrete time by the positions x_k^q, and applies control actions u_k^q, with $k \in \mathbb{Z}_+$ indexing the time step. For convenience, the motion dynamics of all the robots are the same, defined as $g : X \times U \to X$:

$$x_{k+1}^q = g(x_k^q, u_k^q), \forall q, k, \tag{2}$$

and robots share the same sampling time T_s. Note that dynamics g will be first-order in this work, but our method may be generalized to more complex – e.g. second-order – models like in (Lalish and Morgansen, 2008; Plaku et al., 2010), where dynamics additionally include states like velocities and torques.

Remark 3. *Denote the velocity of robot q at step k by v_k^q, and the maximum velocity by $\overline{v} \geq v_k^q$, $\forall q, k$. Then, the maximum step length of any robot, denoted by $\overline{s}$, is $\overline{s} = \overline{v}T_s$.*

The following reachability assumption holds for all the robots. Note that the q index is omitted here in order to simplify notation.

Assumption 2. ***Reachability:*** *$\exists R > 0$ such that $\forall x \in X$ and $\forall x' \in \mathbb{B}(x, R)$, $\exists u \in U$ such that $g(x, u) = x'$, where $\mathbb{B}(x, R)$ is the ball centered at x and of radius R.*

Assumption 2 ensures that the robots can eventually reach all points within the set X. Such an assumption is not uncommon and was made e.g. for the source seeking methods of Azuma et al. (2012); Khong et al. (2014b); Zou et al. (2015).

Assumption 3. ***Sampling and localization:*** *Robots can only sample the objective function at their current position, and have access to their absolute position.*

It is reasonable to assume that robots are equipped with sensors enabling the evaluation of $f(x_k^q)$ (which we call "sampling"), only at their current positions $x_k^q, \forall q, k$. Additionally, robots can determine their absolute position coordinates x_k^q in many situations, e.g. during operation in a GPS-enabled field (Khong et al., 2014b; Zou et al., 2015; Gronemeyer et al., 2017). The results of the following sections hold under Assumptions 1-3.

To address the multirobot problem defined above, the method proposed in this work adapts the global optimizer Simultaneous Optimistic Optimization (SOO) (Munos, 2011, 2014), which belongs to the broader class of divide-the-best algorithms (Sergeyev, 1998). SOO aims to find all global optima of a Lipschitz-continuous function over a compact set, without knowledge of the underlying Lipschitz constant. This set is incrementally partitioned into a hierarchical tree structure, where each leaf node (cell) is associated with its function value (evaluated at the cell's center) and a depth-dependent diameter (the maximum distance from the cell center to its boundary, called cell size, scaled by the unknown Lipschitz constant). Nodes are chosen for further expansion based on their combination of function value and depth in the tree, since computing diameters is not possible without knowledge of the Lipschitz constant. At each iteration, SOO expands at each depth one largest-function-value node, but only if its value is also larger than those of all nodes at shallower depths. This condition is imposed because if it were not true, then some shallower node would have both a larger cell size (because cells are larger higher in the tree) and a larger sample value, so for any Lipschitz constant, its upper bound would be larger. Any node satisfying the condition may have the largest upper bound (and hence good chances of containing an optimum) for some Lipschitz constant. SOO guarantees convergence at well-characterized rates to all global optima of the objective function.

3. VSOO algorithm

The method proposed in this paper is called Voronoi Simultaneous Optimistic Optimization (VSOO). It adapts the SOO approach of Munos (2014) to the MR-PAO setting: mobile robots simultaneously search for the global maxima of a Lipschitz-continuous function, whose Lipschitz constant is unknown. The robots sequentially split the search space into increasingly finer partitions that lead closer to all x^* in X. Different from SOO, due to the motion constraints of each robot (e.g. limited velocity), the next-step samples are limited to a neighborhood of the robot's current position. This makes the sampling strategy in SOO inappropriate, as it would require sampling arbitrarily far away states at consecutive steps. To address this issue, VSOO replaces the hierarchical partitioning with Voronoi partitioning, as will be explained next.

Given a time step k, let S_k be the set of states (positions) sampled by all robots in VSOO up to that point, and X_i^k be the Voronoi cells constructed using as centers the states $x_i \in S_k$. Each index i is uniquely allocated once x_i is sampled, and remains unchanged throughout the algorithm. Thus, at any k, x_i remains the center of the (changing over time) Voronoi cell X_i^k. Denote by V_i^k the set of vertices corresponding to X_i^k, $\forall i, k$. Then, $\cup_i X_i^k = X$, and sets V_i^k are not disjoint, as adjacent cells will have common sides, but their interiors are disjoint.

Each cell X_i^k is characterized by its underlying function value $f(x_i)$, and a cell size δ_i^k defined as follows:

$$\delta_i^k = \max_j \{\|v_{i,j} - x_i\| \mid v_{i,j} \in V_i^k\} = \max_{x \in X_i^k} \|x - x_i\|. \qquad (3)$$

Cell X_i^k is dominated by $X_{i'}^k$ if both its function value and cell size are strictly less than those of $X_{i'}^k$, i.e. $f(x_{i'}) > f(x_i)$ and $\delta_{i'}^k > \delta_i^k$.

The domination concept is important in VSOO for the following reason. If the Lipschitz constant M were available to the algorithm, cells could have been ranked for expansion by sorting descendingly by *upper bounds* defined as $f(x_i) + M\delta_i^k$ for each cell X_i^k, like in Deterministic Optimistic Optimization (Munos, 2014) or Path-Aware Optimistic Optimization (Sântejudean et al., 2025). By expanding largest-upper bound cells, convergence to all global optima would be ensured per Munos (2014). However, VSOO cannot use this bound-based ranking system since it does not have access to M. Instead, all undominated cells must be considered as worth expanding, as for these cells there exists an M such that their bound would be maximal. We can nevertheless eliminate from consideration any dominated cell X_i^k since it has an upper bound smaller than that of any $X_{i'}^k$ dominating it, no matter the value of M (as $f(x_{i'}) > f(x_i)$ and $\delta_{i'}^k > \delta_i^k$). Note that even if X_i^k is dominated at step k, it may eventually become undominated, e.g. once another cell is expanded to a smaller cell size than δ_i^k. Finally, even though the Lipschitz constant is not explicitly used, the objective f must still be Lipschitz continuous for the above procedure to work.

Remark 4. *Since our problem involves mobile robots, VSOO will be studied in dimensions $n \in \{1, 2, 3\}$, as in single or multi-robot source seeking (Khong et al., 2014b; Matveev et al., 2014; Gronemeyer et al., 2017). Nevertheless, ideas on how to extend this method to higher dimensions will also be provided.*

We will first explain how robots select Voronoi cells for expansion in Section 3.1, followed by a description of the procedure for cell expansion in Section 3.2. Then, in Section 3.3, we explain how these elements are joined together in the overall VSOO algorithm.

3.1. Cell selection and types of robots in VSOO

The VSOO method employs two different types of mobile robots – *exploration* and *exploitation* robots – when searching for x^*. The type of robot does not impose constraints on its dynamics; exploration and exploitation robots may otherwise be identical. If the robots are different, it is also possible to give e.g. faster robots the exploration task.

We consider l_1 exploration robots tasked with the expansion of cells having the largest cell size among all undominated cells X_i^k, where $1 \leq l_1 < p$ is fixed throughout the algorithm. For this, define the array $\boldsymbol{X}^k$ which contains all the undominated cells sorted first descendingly by cell size δ_i^k, and then by function value $f(x_i)$. Exploration robots expand the first l_1 highest ranking cells in $\boldsymbol{X}^k$ (one per robot). At least one robot should be of exploration type. Exploration robots will create a progressively finer, nearly uniform partitioning of X that eventually leads arbitrarily close to all $x \in X$ (and, consequently, to all $x^* \in X^*$), thereby ensuring everywhere-dense and global convergence, as will be proven in Theorem 1 of Section 4.

The remaining $l_2 = p - l_1$ robots are of exploitation type, tasked with the expansion of Voronoi cells located in regions where large samples were previously acquired. The expectation is that (global) maxima are located in high f-valued regions. Exploitation robots choose to expand the highest-ranking cells from a second array $\boldsymbol{X}^{k*}$, built by sorting all the undominated Voronoi cells descendingly by their function values $f(x_i)$, and then by their cell size δ_i^k.

A possible issue for the exploitation robots occurs when large samples have only been taken near one maximum, while other high-valued regions were not yet visited. In this case, most – or all – robots choosing cells in $\boldsymbol{X}^{k*}$ may focus their search in the neighborhood of that particular maximum, disregarding entirely other regions containing maxima. To avoid this, we impose that the centers of the newly chosen cells in $\boldsymbol{X}^{k*}$ are at least at a given distance away from all the cell centers currently selected for expansion. This distance will be called the exclusion distance and will be denoted by σ.

The use of the exclusion zone comes with a downside. If multiple global optima are located in a ball $\mathbb{B}(x^*, \sigma)$ inside which an exploitation robot q is currently expanding a cell, cells containing other maxima in $\mathbb{B}(x^*, \sigma)$ might be excluded for expansion by exploitation robots other than q due to σ. This means that some maxima in $\mathbb{B}(x^*, \sigma)$ will be found by exploitation robots at best with σ-accuracy. However, convergence towards these global maxima is still ensured by the exploration robots that will eventually sample ε-close to all $x^* \in X^*$, for any $\varepsilon > 0$. In any case, it is advisable to choose σ significantly smaller than the size of the search space X to prevent the exclusion of too many elements in X^* from the search conducted by the exploitation robots.

At certain steps k, tie-breaking may be required between $q' > 1$ robots of the same type that need to simultaneously select cells for expansion. In such a case, each robot will choose a cell among the q' highest-ranking cells in $\boldsymbol{X}^k$ (for exploration) or $\boldsymbol{X}^{k*}$ (for exploitation), that was not yet chosen for expansion by another robot, and is closest relative to its current position. When cells are selected from $\boldsymbol{X}^{k*}$, the σ distance criterion between the cell centers will also be applied.

3.2. Voronoi cell expansion

Let X_i^k be a cell targeted for expansion by robot q at step k. For the expansion of this cell to be completed, the robot must sample a total of $2n$ points on its frontier, forming a "cross-like" symbol in n dimensions; see Figure 1 left plot for some intuition. These points are referred to as expansion points, and collectively constitute the array of targets T^q, as elements $x_t^q \in T^q$ sequentially become targets of robot q.

The first expansion point (end of the cross) in T^q is $\overline{x}_i \in \arg\max_{v_{i,j} \in V_i^k} \|v_{i,j} - x_i\|$, namely a vertex farthest from x_i, which also dictates cell size in (3). The second point at the opposite end of the cross, is the intersection of the line passing through $\overline{x}_i$ and center x_i with the cell frontier. The other elements in T^q are obtained by taking the remaining $n-1$ perpendiculars that together with the line created by the first two points form an orthogonal coordinate system (the n-dimensional cross), and intersecting these perpendiculars with the frontier of the Voronoi cell. Thus, a total of $2n$ samples will be acquired for the expansion of an n-dimensional Voronoi cell; in particular, for $n \in \{1, 2, 3\}$, the number of expansion points is $2, 4$ or 6, respectively.

The cross-like cell expansion was chosen because, once all expansion points of X_i^k have been sampled, it ensures a contraction of the cell size. More details on this contraction will be given later in Section 4.

The left plot in Figure 1 provides an example of a 2D Voronoi partitioning in which cell X_i^k was chosen for expansion. Cell frontiers are marked with red lines, and cell X_i^k is also delimited by the blue dashed-lines drawn on top of the red ones. Cell centers (including x_i of X_i^k) are marked with black circles, and elements of T^q (expansion points) with black stars. The vertex that gives the cell size δ_i^k of X_i^k is the first element in T^q, denoted by $\overline{x}_i$ (upper-right expansion point). The second expansion point is at the intersection on the opposite side of the line given by the points x_i and $\overline{x}_i$. The perpendicular on this line, passing through the cell center x_i intersects the Voronoi frontier in the last two expansion points. The total number of points is, therefore, $2n = 4$. Once all expansion points were successfully sampled, becoming thus cell centers at future iterations, the new Voronoi partitioning is shown on the right plot of Figure 1. To enhance figure readability, intermediate samples acquired between expansion points have been omitted; in reality, all samples are used by VSOO when partitioning the search space.

3.3. Overall VSOO method

The steps of VSOO are summarized in Algorithm 1, and the flowchart depicted later in Figure 12 of Section 6. The algorithm receives as input the search space X, the motion dynamics g, the number of exploration robots l_1 and exploitation robots

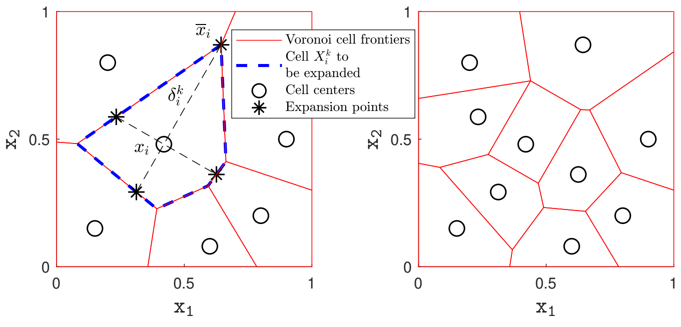


Figure 1: Left: Initial Voronoi partitioning and illustration of a cell expansion procedure. Right: Partitioning after the cell expansion occurred (i.e. once all expansion points have been sampled). Intermediate samples between expansion points are omitted for clarity.

l_2, the exclusion distance σ, the total number of iterations N and the gap weight ζ.

Algorithm 1 VSOO

```
Input: search space $X$, dynamics $g$, numbers of exploration and exploitation robots $l_1, l_2$, exclusion distance $\sigma$, maximum number of iterations $N$, gap weight $\zeta$
1: allocate any $l_1$ robots to exploration and the remaining $l_2$ to exploitation
2: initialize robots in states $x_0^q$, $\forall q = 1, \dots, p$
3: set initial robot targets $x_t^q = x_k^q$, initial target arrays $T^q = \{x_t^q\}$, and set of samples $S_0 \leftarrow \emptyset$
4: for each step $k = 0, \dots, N-1$ do
5:     take samples $f(x_k^q)$, $\forall q$, and add $(x_k^q, f(x_k^q))$ to $S_k$
6:     partition $X$ into Voronoi cells using the states in $S_k$
7:     for each robot $q = 1, \dots, p$ do
8:         if $x_k^q = x_t^q$ then
9:             drop the visited expansion point $x_t^q$ from
                   targets array $T^q$, $T^q = T^q \setminus \{x_t^q\}$
10:            if $T^q = \emptyset$ then
11:                update $T^q$ with rules in Sections 3.1, 3.2
12:            end if
13:            select next target $x_t^q = \arg\min_{x \in T^q} \|x - x_k^q\|$
14:        end if
15:        find $u_k^q = \arg\min_u \|x_t^q - g(x_k^q, u)\|$, then apply $u_k^q$
               to reach $x_{k+1}^q$
16:    end for
17:    $S_{k+1} = S_k$
18: end for
Output: $\overline{f_N} = \max_{x_s \in S_N} f(x_s)$ and $\widehat{X_\zeta^*} = \{x_s \in S_N \,|\, \overline{f_N} - f(x_s) \le \zeta(\overline{f_N} - \underline{f_N})\}$.
```

VSOO works as follows. Each robot q starts in the initial state x_0^q, which is also taken as the robot's first target x_t^q, i.e. $T^q = \{x_0^q\}$, so as to properly trigger the first round of cell selection. At every algorithm step k, where k indexes both the current iteration of VSOO and the trajectory step for any robot q, all robots sample their current positions x_k^q and add the pairs $(x_k^q, f(x_k^q))$ to the set S_k. The search space X is then partitioned into Voronoi cells using as centers the sampled states in S_k.

Next, each robot is checking if it has reached the previously targeted state x_t^q. If so, that state is dropped from the targets array T^q. When this array becomes empty, i.e. at $k = 0$ or when the samples needed to fully expand a Voronoi cell per the procedure in the previous section have all been visited, a new Voronoi cell is chosen for expansion based on the robot's type (exploration/exploitation) and the rules in Sections 3.1 and 3.2. The resulting expansion points are stored in the targets array T^q, and x_t^q is set as the closest element in T^q relative to robot's current position, i.e. $x_t^q = \arg\min_{x \in T^q} \|x - x_k^q\|$. Then, the action u_k^q leading the robot closest to its current target is determined and applied so that the robot reaches x_{k+1}^q. Note that the new sample set S_{k+1} includes all of the previous samples and the ones that are acquired at the $k + 1^{\text{th}}$ iteration.

The algorithm stops when the number of VSOO iterations (trajectory steps per robot) has been exhausted, at which point the method returns the largest objective value seen so far $\overline{f_N} = \max_{x_s \in S_N} f(x_s)$. The algorithm also returns $\widehat{X_\zeta^*}$, an approximate set of near-optimal points in X driven by parameter $\zeta \in (0, 1)$. This set is defined as:

$$\widehat{X_\zeta^*} = \{x_s \in S_k \,|\, \overline{f_k} - f(x_s) \le \zeta(\overline{f_k} - \underline{f_k})\}, \tag{4}$$

where ζ is a weight applied to the gap between the largest sample $\overline{f_N}$, and the lowest sample $\underline{f_N} = \min_{x_s \in S_N} f(x_s)$. The larger the weight ζ, the wider the approximation of the near-optimal points $\widehat{X_\zeta^*}$ returned by VSOO.

4. Convergence analysis

We prove the everywhere-dense and global convergence of VSOO in Section 4.1. In Section 4.2, we show contraction factors that characterize a systematic reduction in the size of a cell after its expansion, which are then used in Section 4.3 to derive convergence rates for some representative shapes of the objective function.

4.1. Asymptotic convergence

Theorem 1. ***Everywhere-dense and global convergence of VSOO:*** *For any $\varepsilon > 0$, there exists a large enough number k of VSOO iterations such that $\forall x \in X$, $\exists x_s \in S_k$ with $\|x - x_s\| < \varepsilon$. In other words, VSOO will eventually sample arbitrarily close to each $x \in X$, consequently converging towards all global maxima in X.*

Proof. Suppose the contrary: $\exists x \in X$ and $\varepsilon > 0$ such that $\min_{x_s \in S_k} \|x - x_s\| \ge \varepsilon$, $\forall k$, i.e. no samples in S_k will ever get closer than ε to x. Thus, Voronoi cells containing x will always have the cell size greater than or equal to ε, which leads to:

$$\varepsilon \le \min_{x_s \in S_k} \|x - x_s\| \le \delta_s^k \le \overline{\delta}^k, \forall k, \tag{5}$$

where $\overline{\delta}^k = \max_i \delta_i^k$ is the maximum size of all the cells comprising the partitioning of X at step k. Recall that at least one exploration robot is always expanding cells corresponding to $\overline{\delta}^k$ in $\boldsymbol{X}^k$. To complete the expansion process, that robot will sample the vertex $\overline{x}_t$ corresponding to $\overline{\delta}^k$. According to (5), future such targets $\overline{x}_t^+$ will always have to be at least ε away from previous ones and all other samples in S_k. This translates to:

$$\varepsilon \leq \min_{x_s \in S_k} \|\overline{x}_t^+ - x_s\|, \forall k. \tag{6}$$

The number of future targets $\overline{x}_t^+$ that must maintain per condition (6) an ε-distance from all samples in S_k (which includes previous targets) will inevitably grow unbounded over time within the compact search space X. However, there exists a finite limit on the number of disjoint spheres with a radius of at least ε and centers in S_k that can fit inside X – related to the sphere packing number (Hales, 2011; Conway and Sloane, 2013). Any new target added to S_k after this number is reached will imply the existence of at least one sample $x_s \in S_k$ s.t. $\|\overline{x}_t^+ - x_s\| < \varepsilon$, which is in contradiction with (6). Therefore, the everywhere-dense and global convergence of VSOO was proven. □

Remark 5. *The sampling strategy which considers the cell diameter as the characteristic after which the "best" cell is chosen for expansion (division) can be classified as a divide-the-best algorithm (Sergeyev, 1998). Thus, the everywhere-dense and global convergence also follows from Sergeyev (1998). We nevertheless provide our own proof because it offers insight into the Voronoi cell expansion performed by VSOO.*

4.2. Contractions following cell expansions

A contraction factor $\gamma \in (0, 1)$ is sought that characterizes a systematic reduction of the cell size δ_i^k of X_i^k after it was fully expanded at some future step k', i.e. for $X_i^{k'}$ centered in the same x_i and having size $\delta_i^{k'}$, we wish $\delta_i^{k'} < \gamma\delta_i^k$. This contraction factor will be later used to derive VSOO convergence rates. Given that this study focuses on one-, two-, and three-dimensional search spaces, the contraction factor will be provided for these cases. An idea on how to expand this contraction property to higher dimensions will also be provided.

Theorem 2. ***Contraction of the cell size in* 1 – 3*D:*** *Let X_i^k be a cell with size δ_i^k targeted for expansion by robot q at step k. At the future step k' (which is finite since X is compact and robots commit to sampling all expansion points) so that all expansion points $T^q = \{x_{t,j}^q | j = 1, \dots, 2n\}$ of X_i^k have been sampled, the following holds:*

$$\min_{x_s \in S_{k'}} \|x - x_s\| \leq \gamma\delta_i^k, \forall x \in X_i^k, \tag{7}$$

where $\gamma \in (0, 1)$ is the contraction factor. Specifically, when the dimensionality of the space is $n = 1$, $\gamma = \frac{1}{2}$, for $n = 2$, $\gamma = \sqrt{2 - \sqrt{2}}$, and for $n = 3$, $\gamma = \sqrt{2 - \frac{2\sqrt{3}}{3}}$.

Proof. To simplify the proof line without restricting its generality, intermediate samples between the expansion points in T^q, and samples acquired from step k to k' by robots other than q will be disregarded. In other words, we only add points T^q to the future set of samples $S_{k'}$ that we characterize. Indeed, if (7) holds for $S_{k'} = S_k \cup T^q$, adding also the intermediate samples between states in T^q and/or samples gathered by other robots, can only lead to even smaller distances in (7), i.e. smaller contraction factors. Thus, the value of γ proven here can be seen as a worst case.

Since cell X_i^k belongs to the Voronoi partition of X built at step k, initially it follows that:

$$\min_{x_s \in S_k} \|x' - x_s\| = \|x' - x_i\| \leq \delta_i^k, \forall x' \in X_i^k. \tag{8}$$

1D: For $n = 1$, expansion points are taken at the ends of the intervals selected for expansion (as Voronoi cells reduce to intervals in 1D), which quickly leads to $|T^q| = 2n = 2$ and $\gamma = \frac{1}{2}$.

To reach a contraction in (8) so that (7) holds in dimensions 2 and 3, two distinct cases are possible:

1. $x \in X_i^{k'}$, i.e. contraction of cell size δ_i^k to $\delta_i^{k'}$ of $X_i^{k'}$,
2. $x \in X_i^k \setminus X_i^{k'} = X_i^k \cap X_j^{k'}$, i.e. contraction of δ_i^k when x belongs to the intersection of X_i^k with one of the newly created cells $X_j^{k'}$ with centers $x_{t,j}^q \in T^q$ chosen at step k (expansion points indexed by j, for ease of reference).

2D case 1: Figure 2 shows a corner of the Voronoi cell X_i^k with center x_i, cell size δ_i^k and continuous-line red frontier, markings consistent with Figure 1. Expansion point $x_{t,1}^q$ is the vertex corresponding to δ_i^k, while $x_{t,2}^q$ is another expansion point adjacent to it. After sampling the four points in T^q (set at step k), the newly created Voronoi cells with centers in T^q would surround the new cell $X_i^{k'}$, whose center remains x_i. A few of the new frontiers drawn at step k' are marked in the same figure with red dashed lines.

The cardinality of T^q in 2D is $2n = 4$, and the vertices of $X_i^{k'}$ are the intersection points of the perpendicular bisectors of the segments connecting the endpoint pairs $(x_i, x_{t,j}^q)$. Thus, the shape of $X_i^{k'}$ will be a rectangle with its side length at most δ_i^k (see Figure 2 for some intuition). The cell size of $X_i^{k'}$ is at most $\frac{\delta_i^k\sqrt{2}}{2}$, and the contraction in case 1 is:

$$\gamma_1 := \frac{\|x_i - v'_{i,1}\|}{\delta_i^k} = \frac{\sqrt{2}}{2}, \tag{9}$$

where $v'_{i,1}$ is the common vertex shared by the cells with the centers x_i, $x_{t,1}^q$ and $x_{t,2}^q$.

2D case 2: When $x \in X_i^k \cap X_j^{k'}$, $\arg\max_j \|x - x_{t,j}^q\|$ can only contain (a) vertices shared by $X_j^{k'}$ with its adjacent cells (cells with centers in T^q or the vertices of $X_i^{k'}$, e.g., $v'_{i,1}$ in Figure 2); or (b) vertices of X_i^k that are closest to $x_{t,j}^q$ (e.g., $v_{i,1}$ in the same figure). In situation (a), the shared vertex denoted by $v'_{i,j}$, is also part of $X_j^{k'}$, i.e.:

$$\|v'_{i,j} - x_{t,j}^q\| = \|v'_{i,j} - x_i\| \leq \frac{\delta_i^k\sqrt{2}}{2}, \tag{10}$$

which leads to a contraction of $\frac{\sqrt{2}}{2}$. In situation (b), the largest distance is obtained when $v_{i,j}$ is located as far away from the

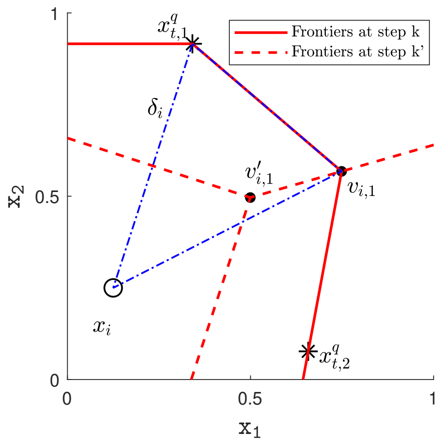


Figure 2: Illustration of a cell size contraction in 2D after sampling expansion points $x_{t,1}$ and $x_{t,2}$.

center of X_i^k as possible, i.e. $\|v_{i,j} - x_i\| = \delta_i^k$. Using the cosine theorem in the triangle with the vertices x_i, $v_{i,j}$ and $x^q_{t,j}$ (e.g. triangle with vertices $x_i, x^q_{t,1}, v_{i,1}$, marked with dashed-dotted blue lines in Figure 2), it follows that:

$$\|v_{i,j} - x^q_{t,j}\| = \sqrt{2-\sqrt{2}}\delta_i^k, \tag{11}$$

which leads to a contraction factor of:

$$\gamma_2 := \frac{\|v_{i,j} - x^q_{t,j}\|}{\delta_i^k} = \sqrt{2-\sqrt{2}}. \tag{12}$$

Therefore, given all the cases studied above, a contraction factor that always holds is:

$$\gamma = \max(\gamma_1, \gamma_2) = \gamma_2 = \sqrt{2-\sqrt{2}} \approx 0.7654. \tag{13}$$

3D case 1: The expansion of cell X_i^k requires $2n = 6$ expansion points to be sampled by robot q. Thus, $|T^q| = 6$, and the shape of the resulting cell $X_i^{k'}$ will be a 3-dimensional rectangular parallelepiped with side length at most δ_i^k. Half of the parallelepiped's diagonal is $\frac{\delta_i^k\sqrt{3}}{2}$, which leads to $\gamma_1 = \frac{\sqrt{3}}{2}$.

3D case 2: The approach for computing γ_2 is similar to the case taken in 2D, and it is not difficult to prove that $\|x^q_{t,j} - v_{i,j}\| \le \sqrt{2 - \frac{2\sqrt{3}}{3}}\delta_i^k$. Then, $\gamma_2 = \sqrt{2-\frac{2\sqrt{3}}{3}}$, and a worst-case contraction factor could be taken as $\gamma = \gamma_2 \approx 0.9194$. □

Remark 6. *It is apparent that for dimensions $n > 3$, sampling only the ends of the n-dimensional cross will not always ensure a contraction, i.e. $\gamma < 1$. Indeed, following the same proof line from Theorem 2, for $n = 4$, it follows that $\gamma_1 = \frac{\sqrt{4}}{2} = 1$. To tackle this, one can split the segments $(x_i, x^q_{t,j})$ connecting the cell center to the expansion points, considered above in the proof of Theorem 2, in several equal parts across each cross axis (as many as needed to get $\gamma < 1$).*

4.3. Asymptotic convergence rates in interesting special cases

This section presents our results on VSOO convergence rates, characterizing the algorithm's complexity in terms of the number of iterations (or equivalently, number of steps/samples for each robot) required to reach ε−accuracy w.r.t. all maxima in X^*. Rates will be studied for two particular cases: one in which f is constant (flat) over X, and a second in which f decreases uniformly radially around all its global maxima. These two cases are representative because they are respectively the most difficult and the easiest problems for VSOO. Note that rates hold for a large number of iterations, given their asymptotic nature.

The following result will be presented first, as it is useful for deriving the convergence rates.

Lemma 1. *Fix $\tau \in \mathbb{R}$ such that $0 < \tau < \delta_X$, where $\delta_X := \max_{x,x'\in X}\|x - x'\|$ denotes the size of X. The maximum number of iterations k_τ required by VSOO to reach τ-close to all points in X (i.e., so that $\forall x \in X$, $\exists x_s \in S_{k_\tau}$ with $\|x - x_s\| < \tau$) is no larger than $2n\overline{N}(\overline{N}p(1 + 2n) + 1)^{\lceil \log_\gamma \frac{\tau}{\delta_X} \rceil}$, where $\overline{N}$ denotes the maximum number of steps a robot requires to travel between any two points in X.*

Proof. To reach τ-closeness to all points in X it suffices if all cells in X have a size no larger than τ. Since exploration robots expand largest-size cells, we focus on them and consider the case when $l_1 = 1$, i.e. there is only one exploration robot. Recall that the number of expansion points a robot has to sample to expand a cell is $2n$. A robot requires no more than $\overline{N} + 2n\overline{N} = \overline{N}(1 + 2n)$ steps to travel towards (in $\overline{N}$ steps) and expand (in $2n\overline{N}$ steps) a new targeted cell at any VSOO iteration.

Initially, each robot generates its own Voronoi cell centered at its position, at the initial sample it takes. After no more than $2n\overline{N}$ VSOO iterations/steps per robot these initial p cells are fully expanded, leading to a contraction with factor γ of the size of X. This means that the size of the largest cell in X is no larger than $\gamma\delta_X$. Since each robot step leads to a sample, and each sample could be the center of a newly created cell, at most $2n\overline{N}p$ cells will comprise the new partitioning of X.

To apply another contraction and reach $\gamma^2\delta_X$-close to all points in X, at most $2n\overline{N}p$ additional cells have to be expanded. At worst, all these cells are be expanded by a single exploration robot, leading to at most $\overline{N}(1+2n)\cdot 2n\overline{N}p = 2n\overline{N}(\overline{N}p(1+2n))$ additional steps per robot/VSOO iterations. Accounting also for the previous $2n\overline{N}$ iterations, reaching a $\gamma^2\delta_X$-closeness requires no more than $2n\overline{N}(\overline{N}p(1+2n)+1)$ total iterations, corresponding to at most $2n\overline{N}(\overline{N}p(1 + 2n) + 1)p$ robot samples or cells making up the partitioning of X. We have obtained the base case of the induction that will be proven next.

For any $h \in \mathbb{Z}^*_+$, start the induction step by the following statement: VSOO has reached $\gamma^h\delta_X$−closeness to all points in X after no more than $2n\overline{N}(\overline{N}p(1 + 2n) + 1)^{h-1}$ iterations and $2n\overline{N}(\overline{N}p(1 + 2n) + 1)^{h-1}p$ robot samples. Thus, there are at most $2n\overline{N}(\overline{N}p(1 + 2n) + 1)^{h-1}p$ cells making up the partitioning of X. We suppose this statement holds for h, and to obtain our induction step we prove it for $h + 1$.

To apply another contraction and reach $\gamma^{h+1}\delta_X$-close to all points in X, at most $2n\overline{N}(\overline{N}p(1 + 2n) + 1)^{h-1}p$ additional cells have to be expanded. At worst, all these cells would have to be expanded by a single exploration robot, leading to at most $\overline{N}(1 + 2n) \cdot 2n\overline{N}(\overline{N}p(1 + 2n) + 1)^{h-1}p$ additional steps per robot/VSOO iterations. Cumulating with the previous $2n\overline{N}(\overline{N}p(1+2n)+1)^{h-1}$ iterations, reaching $\gamma^{h+1}\delta_X$-closeness requires no more than $2n\overline{N}(\overline{N}p(1+2n)+1)^h$ total iterations, which leads to at most $2n\overline{N}(\overline{N}p(1+2n)+1)^h p$ robot samples, or equivalently, cells making up the partitioning of X. This proves our desired property for $h+1$.

Finally, by taking h such that $\tau \geq \gamma^{h+1}\delta_X$, e.g., $h = \lceil \log_\gamma \frac{\tau}{\delta_X} \rceil$, the proof is complete. □

Remark 7. *While conservative, the upper bound on k_τ given in Lemma 1 depends only on τ and the finite constants n, $\overline{N}$, p, γ, and δ_X. Next, we will apply the bound with finite (not infinitesimal) values of τ, which will allow us to integrate it in the constant part of the asymptotic convergence rates.*

Theorem 3. ***Convergence rate of VSOO in the "flat" case:*** *Take $f(x) = C$, $\forall x \in X$. Let $\gamma \in (0,1)$ be the contraction factor of Theorem 2, and $\varepsilon > 0$ a small positive constant. For a small enough exclusion zone σ_ε, the number of steps for each robot required to reach ε-close to all optima of f — i.e., to ensure that $\exists x_s \in S_k$ with $\|x - x_s\| < \varepsilon$, $\forall x \in X = X^*$ — is at most $O\left(\frac{\mathcal{K}^{\log\varepsilon/\log\gamma}}{p}\right)$, where $\mathcal{K} = 1 + \overline{N} + 2n$.*

Proof. All samples acquired by the robots will have equal value due to f being constant (flat) over X. Thus, according to Section 3.1, exploitation robots will behave like exploration ones, always expanding cells with the largest sizes in X (except for the σ_ε constraint). If the exclusion zone σ_ε is taken small enough, as $\varepsilon \to 0$, no cells from $\boldsymbol{X}^{k^*}$ will be excluded by σ_ε.

Take $\tau = \frac{s}{2}$. By Lemma 1, there exists a finite number $k_\tau \in \mathbb{Z}_+$ of steps for each robot/VSOO iterations after which $\forall x \in X$, $\exists x_s \in S_k$ such that $\|x - x_s\| < \tau = \frac{s}{2}$, $\forall k \geq k_\tau$. Thus, $\overline{\delta}^k < \frac{s}{2}$, and all expansion points in T^q will be reachable within one robot step/VSOO iteration. This means that a robot will require at most $\overline{N} + 2n$ steps to travel towards and expand a new targeted cell for any $k \geq k_\tau$.

Recall the notation $\overline{\delta}^{k_\tau}$ that represents the largest size among the cells comprising the Voronoi partitioning of X at step k_τ. To reach a contraction of factor γ so that all points in X are at most $\gamma\overline{\delta}^{k_\tau}$–away from their closest available sample in S_k, the p robots need to expand at most k_τ cells in X. For this, at most $\frac{k_\tau}{p}\cdot(\overline{N}+2n)$ additional steps for each robot are required, leading to no more than $k_\tau(1+(\overline{N}+2n))$ samples in S_k, or, equivalently, cells in X. To apply another γ contraction and get $\gamma^2\overline{\delta}^{k_\tau}$–close to all points in X, at most $\frac{k_\tau(1+(\overline{N}+2n))}{p}\cdot(\overline{N}+2n)$ additional steps for each robot are needed. At this point, the total number of samples in S_k/cells in X is bounded by $k_\tau(1+\overline{N}+2n)^2$.

Applying the same mechanism up to step $h \in \mathbb{Z}_+$, it follows by induction that to get $\gamma^h\overline{\delta}^{k_\tau}$–close to all points in X, at most $\frac{k_\tau(1+(\overline{N}+2n))^{h-1}}{p}\cdot(\overline{N}+2n)$ additional steps for each robot are required from the previous induction step $h-1$. Take now h such that $\varepsilon \geq \gamma^h\overline{\delta}^{k_\tau}$ (e.g., $h = \lceil \log_\gamma \frac{\varepsilon}{\overline{\delta}^{k_\tau}} \rceil$). Reaching $\gamma^h\overline{\delta}^{k_\tau}$–close to all points in X requires at most N steps for each robot, where:

$$
\begin{aligned}
N &= k_\tau + \sum_{i=1}^{h} \frac{k_\tau(1+\overline{N}+2n)^{h-1}}{p}\cdot(\overline{N}+2n) \\
&= k_\tau + \frac{k_\tau(\overline{N}+2n)}{p}\cdot\frac{(1+\overline{N}+2n)^h - 1}{1+(\overline{N}+2n)-1} \\
&= k_\tau + \frac{k_\tau((1+\overline{N}+2n)^h - 1)}{p} \\
&= O\left(\frac{(1+\overline{N}+2n)^h}{p}\right) = O\left(\frac{\mathcal{K}^h}{p}\right).
\end{aligned}
\tag{14}
$$

Using $\varepsilon \geq \gamma^h\overline{\delta}^{k_\tau}$, from which h can be expressed as:

$$
h \leq \frac{\log\varepsilon - k_\tau\log\overline{\delta}}{\log\gamma}, \tag{15}
$$

it finally follows that reaching ε-optimality requires at most:

$$
N = O\left(\frac{\mathcal{K}^{\frac{\log\varepsilon - k_\tau\log\overline{\delta}}{\log\gamma}}}{p}\right) = O\left(\frac{\mathcal{K}^{\frac{\log\varepsilon}{\log\gamma}}}{p}\right). \tag{16}
$$

steps for each robot. □

Remark 8. *The rates proven in Theorem 3 can be interpreted as follows. The division by p shows that the complexity is smaller when the number of robots increases. The presence of constants $\overline{N}$ and n in $\mathcal{K}$ shows that complexity is larger for larger sizes and dimensionality of the search space. Moreover, we see exponential complexity in $\frac{\log\varepsilon}{\log\gamma}$, which is unavoidable in the flat case.*

In the second convergence rate result, f is required to decrease uniformly radially around all its global maxima (e.g., right circular cones or radial basis functions), so we will call this the "cones" case for easy reference. The "cones" case can indeed be found in practical applications, for instance a team of mobile robots searching for the locations of some wireless antennas (Fink and Kumar, 2010; Buşoniu et al., 2020) or chemical plumes (Spears et al., 2005; He et al., 2019).

Theorem 4. ***Convergence rate of VSOO in the "cones" case:*** *Consider a function f having at most l_2 global maxima in X situated at least σ-away from each other, where l_2 is the number of exploitation robots. Suppose that $\exists\delta_\varepsilon > 0$ such that f decreases uniformly radially around all its global maxima at least inside a ball $\mathbb{B}(x^*, \delta_\varepsilon)$, centered in x^* and of radius δ_ε. Then, the number steps for each robot needed to reach ε–close to all global maxima $x^* \in X^*$ — i.e., to ensure that $\exists x_s \in S_k$ with $\|x^* - x_s\| < \varepsilon, \forall x^*$ — is $O(\frac{\log\varepsilon}{\log\gamma})$.*

Proof. The radially uniform decrease on f around all $x^* \in X^*$ inside $\mathbb{B}(x^*, \delta_\varepsilon)$ is important here, as it ensures that inside $\mathbb{B}(x^*, \delta_\varepsilon)$, the closest sample to x^* also has the largest value. Thus, once an exploitation robot targets to expand a cell fully enclosed in $\mathbb{B}(x^*, \delta_\varepsilon)$ that contains x^*, each following expansion of the same robot will contract the distance to x^* by a factor of γ, provided the robot does not leave $\mathbb{B}(x^*, \delta_\varepsilon)$. The γ-contraction occurs at each expansion since the cells targeted by the exploitation robot will have the largest sample in $\mathbb{B}(x^*, \delta_\varepsilon)$, and due to the uniform radial decrease of f inside $\mathbb{B}(x^*, \delta_\varepsilon)$, will also have the center closest to x^*, thus containing x^* per the Voronoi cell definition.

Since f has at most l_2 global maxima with the properties described above, $\exists \delta'_\varepsilon > 0$ such that $\forall x^* \in X^*$, $\exists \mathbb{B}(x^*, \delta'_\varepsilon)$ so that $\forall x' \in \mathbb{B}(x^*, \delta'_\varepsilon)$:

$$f(x') > f(x), \forall x \in X \setminus \bigcup_{x^* \in X^*} \mathbb{B}(x^*, \delta'_\varepsilon), \tag{17}$$

i.e. any sample acquired by robots outside the union $\cup_{x^* \in X^*} \mathbb{B}(x^*, \delta'_\varepsilon)$ will have a value below those inside the union. Thus, once the exploitation robots have been tasked with the expansion of cells fully enclosed inside the balls $\mathbb{B}(x^*, \delta'_\varepsilon)$, they will not be redirected by larger-valued samples acquired by other robots (as δ'_ε can be taken small enough such that the union of all exclusion zones σ of the exploitation robots fully covers $\cup_{x^* \in X^*} \mathbb{B}(x^*, \delta'_\varepsilon)$).

Take now $\underline{\delta_\varepsilon} = \min\{\delta_\varepsilon, \delta'_\varepsilon, \sigma, \overline{s}\}$. By Lemma 1, $\exists k_\tau \in \mathbb{Z}_+$, $\tau = \frac{\underline{\delta_\varepsilon}}{2}$, a finite number of iterations after which $\forall x^* \in X^*$, $\exists x_s \in S_k$ such that $\|x^* - x_s\| < \tau = \frac{\underline{\delta_\varepsilon}}{2}$, and the cells containing the global maxima are fully enclosed in their corresponding $\mathbb{B}(x^*, \delta'_\varepsilon)$. Thus, at iteration k_τ, each x^* that is at least at σ-distance away from all other global maxima, already has an exploitation robot assigned to it that is currently expanding a cell containing x^*, fully enclosed inside $\mathbb{B}(x^*, \underline{\delta_\varepsilon})$. Following the remarks from the previous two paragraphs, each new cell expansion of each such exploitation robot leads to another contraction with factor γ of the distance to that x^*. Since each expansion point is reachable within one robot step, each cell expansion in this regime requires at most $2n$ steps/iterations.

Finally, $\exists h \in \mathbb{Z}_+$, a number of cell expansions following the k_τ steps for each robot, such that $\varepsilon \geq \gamma^h \underline{\delta_\varepsilon}$. Reaching ε–close to all points in X^* requires at most:

$$N \leq k_\tau + 2nh \tag{18}$$

steps for each robot. This leads to the following bound on the number of steps:

$$N = O\left(k_\tau + 2n\frac{\log\varepsilon - \log\underline{\delta_\varepsilon}}{\log\gamma}\right) = O\left(\frac{\log\varepsilon}{\log\gamma}\right). \tag{19}$$

□

Remark 9. *In the cones case, only the global maxima having their pairwise distances larger than σ are guaranteed to be found with ε-accuracy at a rate $O(\frac{\log\varepsilon}{\log\gamma})$, as these maxima are not excluded from the search of the exploitation robots via their exclusion zones σ. Any maxima that are closer to each other than this limit, as well as any maxima that exceed in number the value of l_2 (the number of exploitation robots), will be reached with ε-accuracy after no more than $O\left(\frac{\mathcal{K}^{\log\varepsilon/\log\gamma}}{l_1}\right)$ steps, per the uniform-exploration rate of Theorem 3 achieved here by the remaining l_1 robots still employed in the exploration of X.*

Remark 10. *Unlike the uniform-exploration rate of Theorem 3, for which the number of steps grew exponentially fast with $\frac{\log\varepsilon}{\log\gamma}$, the cones case shows a logarithmic rate increase w.r.t. the same factor – a very low complexity.*

Overall, our theoretical results proved not just that VSOO achieves everywhere-dense and global convergence, but also supplied convergence rates towards the optima in some interesting special cases. In the flat function case VSOO degraded to the convergence rate of a uniform-sampling algorithm, in which case its guarantees follow from everywhere-dense convergence. However, in the cones case we were able to show a much faster convergence rate. In practice, we expect the algorithm to be somewhere between these two extrema.

5. Numerical results

In the following experiments, multiple simulated robots are considered, with the dynamics (2) instantiated to:

$$x^q_{k+1} = x^q_k + T_s \cdot u^q_{k,1} \cdot [\cos(u^q_{k,2}), \sin(u^q_{k,2})]^T, \forall q, \tag{20}$$

where $T_s = 1$s is the sampling period, and $u^q_k = [u^q_{k,1}, u^q_{k,2}]^T$ contains, respectively, the velocity and the heading of robot q. The velocity is bounded: $u^q_{k,1} \in [0, \overline{v}]$ m/s. The maximum velocity will be chosen directly proportionally to the size of the search space. For a search space $X = [-2, 2] \times [-2, 2]\text{m}^2$, the maximum velocity is set to $\overline{v} = 0.2$m/s, whereas e.g. if X were $[-5, 5] \times [-5, 5]\text{m}^2$, $\overline{v}$ would be 0.5m/s. In this way, we ensure that the performance of the methods is not affected by the size of the search space, and we can more fairly compare results on objective functions that are defined on different domains. The exclusion distance of the exploitation robots is set to $\sigma = \overline{v} \cdot T_s$. Note this value is not sufficiently small to ensure the rates in Theorem 3 in case f is constant (flat), which is not a concern since f will not be flat in the experiments. Finally, the number of robots considered is $p \in \{4, 8, 12\}$, out of which – with the exception of the final simulation – one robot is always exploring and the other are exploiting.

We validate VSOO both on benchmark synthetic optimization functions, as well as on a noisy function that represents a real-life source seeking scenario. The VSOO algorithm will be compared to four baselines, respectively based on Particle Swarm Optimization (PSO) (Kennedy et al., 2001), DIRECT (Jones et al., 1993), Bayes-Swarm-P (Ghassemi et al., 2022), and Bayesian Optimization (BayesOpt) (Frazier, 2018). The first three methods are versions for the source seeking problem that, like VSOO, aim to find the global optima of an unknown objective function with a team of mobile robots with dynamics (2), while the fourth method was adapted by us to the MR-PAO setting. We will describe these methods next.

The PSO method is a population-based, gradient-free stochastic optimization algorithm (Kennedy et al., 2001). The method was adapted for the source seeking problem by Zou et al. (2015), by considering individual robots as particles of a swarm. The robots adapt their path by means of velocity and heading updates at each step, towards a combination of their own best sample and the best sample collected by all robots. The method assumes there is only one global optimum, and given its tendency to be trapped in local optima, the PSO in (Zou et al., 2015) uses an inertia weight to achieve global search. The inertia weight, ω, is updated at each step and depends on a damping factor, λ, that contributes to better global search in the beginning of the simulations, and better local search at the end. Our parameters for this baseline remain consistent with (Zou et al., 2015), i.e. $\omega_{k+1} = \lambda\omega_k$, where $\lambda = 0.95$. However, this addition to classical PSO does not completely eliminate the possibility for the robots to converge to local optima or to only a few of the global optima.

The second baseline is based on DIRECT (Jones et al., 1993), a global optimizer able to search, like SOO, for multiple global optima of a Lipschitz-continuous function whose Lipschitz constant is unknown. DIRECT normalizes the search space to a unit hypercube, identifies at each iteration the potential optimal rectangles that have the largest function values at the centers or have diameters large enough to be good targets for global search, and then samples the centers of these rectangles. The DIRECT method has been adapted to source seeking by Khong et al. (2014b), where the search space is divided into equally-sized regions of responsibility, and each such region has a robot assigned to it. The movement of the robots is restricted to their designated region, so when a potentially optimal rectangle's center falls within a region, only the corresponding robot is able to sample that center. In order to visit all points inside its assigned region in minimum time, each robot solves a traveling salesman problem (TSP), e.g. using the method of Kivelevitch (2024).

The third method is Bayes-Swarm-P (Ghassemi et al., 2022), based on Bayesian Optimization, which uses a Bayesian statistical model surrogate for the objective function that is updated with each newly sampled point; and an acquisition function that determines the future points to be visited. The surrogate is a Gaussian Process (GP) model of the objective. The acquisition function proposed in Ghassemi et al. (2022) balances exploration and exploitation with the robots and reduces the overlap in their planned knowledge gain. This is done by using a local penalizing factor that depends on the knowledge of the maximum function value and on the Lipschitz constant. If these values are not known, they can be approximated, but this increases the number of steps to find the optima. For the following experiments, we used the real maximum function value and an a priori estimate of the Lipschitz constant, so Bayes-Swarm-P has an advantage over VSOO and the other baselines. In addition, the algorithm accounts for range limitations when sharing the samples among robots, but we consider unlimited communication range in order to be fair when comparing it against the other methods. The rest of the parameters were set as suggested by Ghassemi et al. (2022).

The last baseline, BayesOpt, is also based on Bayesian Optimization, in a variant that only requires the objective function to be continuous. This method uses the standard MATLAB `bayesopt` function and its default settings. A GP model is used again, and the acquisition function is of type Expected-Improvement-Per-Second-Plus (Snoek et al., 2012), which calculates the future points to be sampled using the value and the location of the lowest posterior mean given by the GP. Note that this method is designed for minimization, so it will use the negative of the objective functions as they require maximization. We adapted the method to the MR-PAO problem by considering as future targets the best p points determined by the acquisition function that are situated σ-away from each other (recall that p and σ are, respectively, the number of robots and the exclusion distance also used in VSOO). Each robot is assigned the target closest to its current position, and travels towards it with dynamics (2).

For a fair comparison, all methods consider the same initial positions for the robots, with each run of the methods having different initial positions. These positions are generated uniformly randomly, for each robot inside each region of responsibility created by DIRECT. Recall that only DIRECT limits the robots' movement inside their assigned sub-region, while the other methods do not take into account these sub-regions when assigning the robots to their targets.

We first consider benchmark functions with a single global optimum from Gaviano et al. (2003) and Surjanovic and Bingham (2013). These functions have challenging shapes, e.g. the global optimum is situated in areas in which the function's values vary slowly or are surrounded by multiple local optima. The benchmark functions are designed for minimization, whereas in our problem formulation we are searching for maxima, so we will run the tests on the negatives of these functions. For the second part we consider functions with multiple global optima: a function inspired from our previous work Sântejudean et al. (2025), which has three global optima situated in a small area in the search space; and an RSSI field generated using signals collected from Bluetooth Low Energy-enabled devices (BLE), thus simulating a real-robot scenario in which the robots have to find the emission sources of a signal.

To compare the performance of the three methods, we will use a metric that measures how closely the robots' trajectories approach all global optima X^*. Thus, at each kth trajectory step, a distance d_k is computed with:

$$d_k = \frac{\sum_{i=1}^{b} d_{ik}}{b}; d_{ik} = \frac{\sum_{j=1}^{|X^*|} \min\{\|x_j^* - x_s\| | x_j^* \in X^*, x_s \in S_k\}}{|X^*|}, \tag{21}$$

i.e., d_k is the smallest distance from any sample to each maximum, averaged across maxima (note that the number of maxima is finite for all test functions). This metric will be reported against the number of iterations performed by each algorithm, averaged across all experiment runs. In multirobot systems, an iteration signifies a synchronized time step where all robots move and sample once, providing a more accurate representation of real-life effort and aligning better with real time than solely considering the number of function evaluations. We will also compare the execution time of the algorithms. All simulations were performed on an Intel(R) Core(TM) i7-1165G7 (2.80GHz) system with 16 GB RAM, running MATLAB R2024a (except Bayes-Swarm-P which runs in Python, on the same machine).

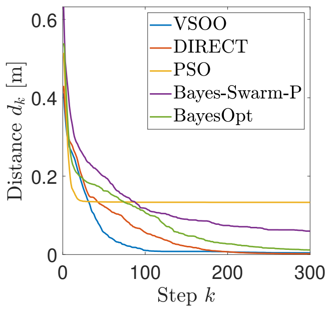
The remainder of the numerical results is organized as follows. Section 5.1 presents the performance and execution time comparison between VSOO, DIRECT, PSO, Bayes-Swarm-P and BayesOpt for $p = 4$ robots on objective functions with single global maximum, in 2 dimensions. In Section 5.2 we validate the quality of $\widehat{X^*_\zeta}$, the approximate set of near-optimal points determined by Algorithm 1, by comparing the distance calculated with points from this set and from all the samples. In this section we also compare VSOO with a naive adaptation of SOO to MR-PAO. Section 5.3 provides the same performance comparison as in the single-global-optimum case, but now for functions that have multiple global optima. All these comparisons are made for a number of $p - 1$ exploitation robots and 1 exploration robot. Lastly, in Section 5.4, we study the performance of VSOO and DIRECT for a varying number of robots, in 3 dimensions. We also discuss the behavior of VSOO when the number of exploration robots is $p - 1$, instead of 1 as it was in the experiments conducted in the previous sections.

5.1. Baseline comparison for single-global-optimum functions

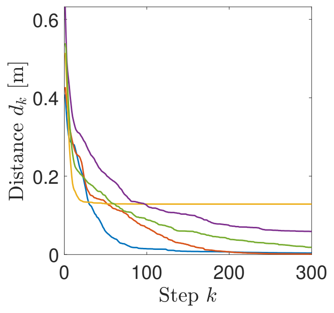
The objective functions considered in this section can be separated into 3 types: continuously differentiable (D-type), twice continuously differentiable (D2-type), and non-differentiable (ND-type). All the functions are concave quadratic and distorted by cubic and quintic polynomials in order to introduce local optima. The software proposed by Gaviano et al. (2003), also known as the GKLS-generator, allows the generation of multiple test functions belonging to the types mentioned above (Sergeyev and Kvasov, 2006; Sergeyev et al., 2018). For each such type, we generated 100 test functions defined on the space $X = [-1, 1] \times [-1, 1]\text{m}^2$, with 10 local optima, keeping the default settings used by Gaviano et al. (2003). The algorithms are run for each function for $p = 4$ robots, and the distance at each step to the global optimum, d_k, is averaged across all functions. Note that the maximum velocity is $\bar{v} = 0.1\text{m/s}$, $\sigma = 0.1\text{m}$, and the number of exploitation robots is $p - 1 = 3$.

Figures 3a-c compare the performance of VSOO with the DIRECT, PSO, Bayes-Swarm-P and BayesOpt baselines on the three types of functions. The comparison remains consistent across all types, showing a better performance in the transient regime for VSOO than for DIRECT, Bayes-Swarm-P and BayesOpt, while PSO demonstrated a propensity for convergence to local optima, reaching the global optimum in only 47%, 56% and 64% of the cases, for the three types of functions respectively. Bayes-Swarm-P did approach the global optimum, but 300 steps were not enough to sufficiently reduce the distance to it. VSOO, DIRECT, and BayesOpt always converged to the global optimum.

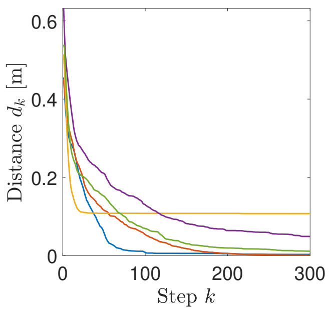
Figure 3d shows the execution time of the algorithms for the D-type functions (execution times remain consistent across all types of functions). VSOO has a linear increase in the execution time as the number of samples increases, unlike DIRECT, Bayes-Swarm-P and BayesOpt which have approximately the same execution time for each step. The average step time of VSOO is 0.01s, faster than of DIRECT's 0.024s and of BayesOpt's of 0.029. Bayes-Swarm-P and PSO are the fastest methods, averaging 0.007s and $6.81 \cdot 10^{-4}$s per step, respectively, but of course they do not find the global optimum.



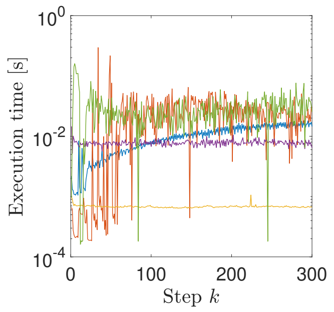
(a) On D-type objective functions



(b) On D2-type objective functions



(c) On ND-type objective functions



(d) Execution times on D-type functions. Note that the vertical axis is logarithmically scaled.

Figure 3: Comparison between VSOO and the baselines in 2D: DIRECT, PSO, Bayes-Swarm-P and BayesOpt. For consistency, the methods are represented using the same colors across all experiments.

5.2. Quality of $\widehat{X^*_\zeta}$ and SOO-like baseline

In the first part of this section, we compare the average distance d_k used in the previous experiments with d'_k, the distance calculated similarly to (21), but using now $x_s \in \widehat{X^*_\zeta}$ and $\zeta = 0.05$, where $\widehat{X^*_\zeta}$ is the approximation of X^* returned by VSOO (Algorithm 1). The objective function is Goldstein-Price, a single global-optimum function chosen to be different from the ones used in Section 5.1, so that we increase the variety of functions. The function is defined on the space $X = [-2, 2] \times [-2, 2]\text{m}^2$, and is given by:

$$\begin{aligned}
&f(x_1, x_2) = \\
&-[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \\
&[30 + (2x_1 - 3x_2)^2(18 - 32x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]. \quad (22)
\end{aligned}$$

Figure 4 (left) shows that the performance of VSOO, when using the distance d'_k, remains similar to the one obtained for d_k, which means in this case $\widehat{X^*_\zeta}$ is a good proxy for the final near-optimal set of samples. We do not report the performance of VSOO and the baselines for this function because all methods behave as before.

Furthermore, for the same function (22), we compare VSOO with a 'naive' adaptation of SOO to MR-PAO. Recall that at each tree depth with unexpanded nodes remaining, SOO expands a largest-value node, unless it is dominated by a shallower one in the tree. Expansion means splitting the node in three equal parts across its largest dimension, and sampling the

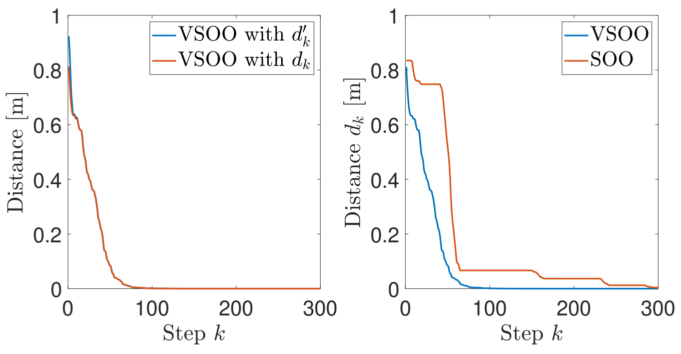


Figure 4: Performance comparison between VSOO with the distance d_k' and d_k (left). Performance comparison between VSOO and naive SOO (right).

centers of these three new children. To adapt SOO, at each iteration the undominated nodes (depths) are equally distributed among the robots (the last robot may have fewer nodes, and in the initial regime there may be robots left with no nodes to expand). A robot responsible for expanding a node travels to each new child's center and reads the function value there. Intermediate samples along the way are ignored during the algorithm, but they are considered when computing the distance d_k, and they are counted on the horizontal axis of Figure 4 (right). In this figure, we see that VSOO significantly outperforms this naive version of SOO.

5.3. *Baseline comparison for multiple-global-optima functions*

The function taken for this case is inspired from our previous work (Sântejudean et al., 2025), defined on $X = [0,4]\times[0,4]\text{m}^2$, and given by:

$$f(x_1, x_2) = \max\{\phi_{1,2,3}(x), \psi_{1,2,3}(x)\}, \qquad (23)$$

where $\phi_i(x) = h_i \cdot \exp\left[\sum_{j=1}^{2} \frac{(x_j - c_{ij})^2}{b_{ij}^2}\right]$ (radial basis functions), and $\psi_i(x) = \lambda_i \cdot [f^* - M'\|x - c_i'\|]$ (cones). The parameters of ϕ are as follows: slope coefficients $\lambda_i = [1, 2/3, 1/2]$, heights $h_i = [255, 255, 127.5]$, widths $b_i = [1.4, 1.4]\lambda_i$, and centers $c_i = [2.75, 3.5]$, $[3.25, 3.25]$, $[3.75, 1.75]$, and of ψ: centers $c_i' = [2.25, 2.25], [1, 0.75], [1.5, 0.5]$, and $M' = 312.5$. Please refer to Figure 5 for a plot of this function. The global optima optima is situated at $x_1^* = [3.25, 3.25]^T$, $x_2^* = [2.25, 2.25]^T$, $x_3^* = [2.75, 3.5]^T$. This function is taken in order to analyze the behavior of the 4 robots when the three optima are all situated in a small region.

Figure 6 shows largely similar trends to the previous experiments in Section 5.1, although the performance gap between VSOO and DIRECT is much larger, BayesOpt improves its performance, and Bayes-Swarm-P performs worse. Whereas VSOO consistently reduces (on average) the distance to all x^*, DIRECT has multiple plateaus in which the distance hardly varies, so it needs more time to reach as close as VSOO to the global optima. The reason for this is that the region in which the multiple global optima are situated corresponds to a single region of responsibility defined by DIRECT. Due to this, only the robot assigned to that region can travel between all these global maxima, which leads to many intermediate steps in the case

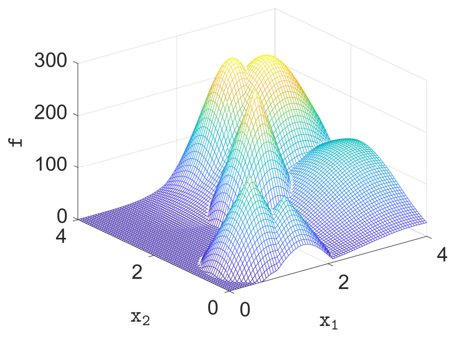


Figure 5: The function adapted from (23) such that all global optima are located inside a single DIRECT sub-region.

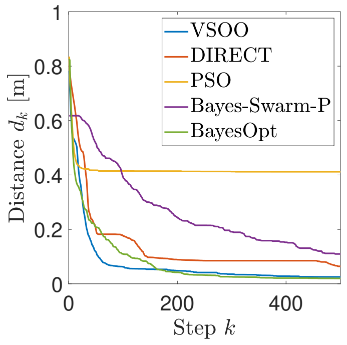


Figure 6: Performance comparison between VSOO and the baselines: DIRECT, PSO, Bayes-Swarm-P and BayesOpt on the function in (23).

of DIRECT. However, this is not the case for VSOO, which allows all robots to focus their search on a single sub-region, if needed, as long as points are further away than σ. Moreover, due to the exclusion distance σ, robots eventually target each optimum separately, no more than one robot per x^*, which greatly helps in decreasing the distance to all optima simultaneously. PSO remains once again suboptimal, discovering x_1^* and x_2^* in 35% of the cases, and x_3^* in only 25% of the cases. Bayes-Swarm-P discovers x_1^* in only 45% of the cases, and x_2^* and x_3^* in 60% of the cases. Due to the exclusion distance and the acquisition function that allow BayesOpt not to overly exploit regions, the average distance to all x^* is considerably reduced for this method. Thus, BayesOpt performance, although worse in the transient regime than of VSOO's, improves afterwards.

For the following experiment, we evaluate VSOO and the baselines on a physical objective function, which is a Received Signal Strength Indication (RSSI) map built using two Bluetooth Low Energy devices (Cypress PSoC4 BLE Pioneer kits). The two antennas were placed at $[-0.9, 0.1]$ and $[-0.9, 1.45]$, inside a search space $X = [-3, 0.2] \times [-0.2, 3]\text{m}^2$. The function was mapped by collecting antenna signals with a ROBOTIS TurtleBot3 equipped with a Raspberry Pi 4 module. This setup is very similar to that used in our real robot experiments in Section 6; see Figure 10 for an overview of the experimental testbed. To build a map for the simulation, the robot moved on an equidistant grid with points spaced at 0.05m on each dimension of X and at each location averaged 100 RSSI samples to estimate the objective value. Note that the robots in VSOO do not sample uniformly like this.

Figure 7 presents on the left a contour plot of the acquired

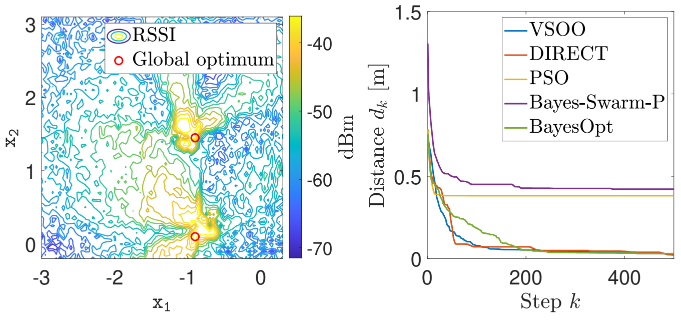


Figure 7: The RSSI map acquired (in dBm) from two Bluetooth Low Energy antennas (Cypress PSoC4 BLE) (left). Performance comparison between VSOO and the baselines: DIRECT, PSO, Bayes-Swarm-P and BayesOpt on the RSSI map (right).

RSSI map on the $3.2 \times 3.2\text{m}^2$ search space X. The location of the global optima (the BLE antennas) is marked on the plot with solid red circles. Although this function is quite noisy, which is true in general for antenna signals (Yılmaz et al., 2013; Wang et al., 2015), the performance comparison between the methods in Figure 7 (right) follows the same trend from above. DIRECT decreases the distance in stairs, in the transient regime, while PSO remains highly suboptimal and only succeeds in finding the first optimum in 35%, and the second one in 20% of the cases. PSO did not discover more than one global optimum in any of the experiments for the functions with multiple global optima. BayesOpt performs as in the previous experiment. However, Bayes-Swarm-P is highly suboptimal, since it finds x_1^* in 35%, and x_2^* in 40% of the cases. We hypothesize that this method is highly affected by the noisy measurements. VSOO is able to find both signal sources as fast as DIRECT, which shows that it handles reasonably well a medium level of noise present on the objective function, although it was not specifically designed for noisy samples.

5.4. *Influence of the ratio of exploration to exploitation robots*

The goal of this section is to study the evolution of VSOO performance when the number of robots in the team varies, as well as when the ratio of exploration to exploitation robots increases. For the experiments in this section, we used the GKLS-generator to determine 100 D-type functions defined on the space $X = [-1, 1] \times [-1, 1] \times [-1, 1]\text{m}^3$, with 10 local optima and one global optimum. In 3D we only compare VSOO and DIRECT. The initial positions of the robots are generated uniformly randomly inside the regions of responsibility defined by DIRECT.

Figure 8 shows that the performance of VSOO improves as the number of robots increases, regardless of the ratio between the exploitation and exploration robots. This is expected, since there are more robots involved in the search, which are therefore able to reach the neighborhoods of the global optima faster. However, we can see that DIRECT does not exhibit a considerably improved performance, since still the optimum belongs to the responsibility region of only one robot. Figure 8 (left) illustrates the case when there are $p-1$ exploitation robots and one exploration robot. This shows the exploitation robots sample around the local optima because the only exploration robot

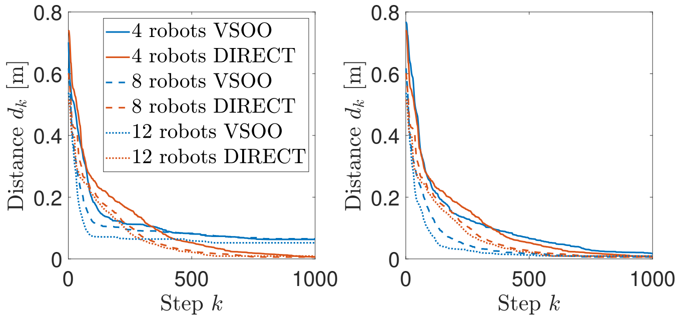


Figure 8: Performance of VSOO and DIRECT when varying the number of robots and the ratio between their types.

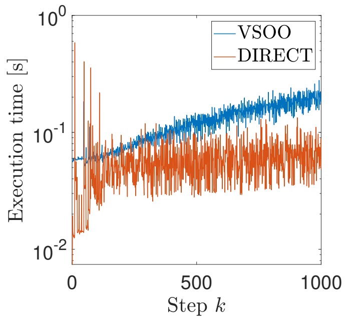


Figure 9: Execution time of VSOO and DIRECT on the D-type functions, in 3D.

needs more steps to discover the area around the global optimum. Figure 8 (right) shows the performance of VSOO when the ratio between exploration and exploitation robots is $p-1$ to 1. Unlike the previous case, the performance is significantly improved, since there are more exploration robots that can reach the area around the global optimum faster, and one exploitation robot is enough to sample around the single global optimum.

We also study the execution times of both methods. Figure 9 shows similar trends for the execution times as in the 2D case, but this time, VSOO is slower than DIRECT, its average step time being 0.143s and of DIRECT 0.056s.

A possible drawback of the method could be the computational cost that might grow faster for larger dimensions compared to the baselines. However, this drawback was not significant in the 2D and 3D cases discussed in this study, where VSOO exhibited competitive execution time. Note that VSOO's execution time is significantly influenced by the efficiency of its Voronoi partitioning. While the current implementation utilizes an incremental approach based on the `voronoiDiagram` function from MATLAB's `Computational Geometry` toolbox, further optimizations may improve computational speed.

Overall, VSOO outperforms the baselines by finding all global optima in fewer steps (on average).

6. Real-robot experimental results

In this section, a real-life implementation of the VSOO algorithm is used to verify that the simulated results also hold in a real-robot scenario. The goal of the experiment is to find a group of radio frequency broadcasting antennas using a team

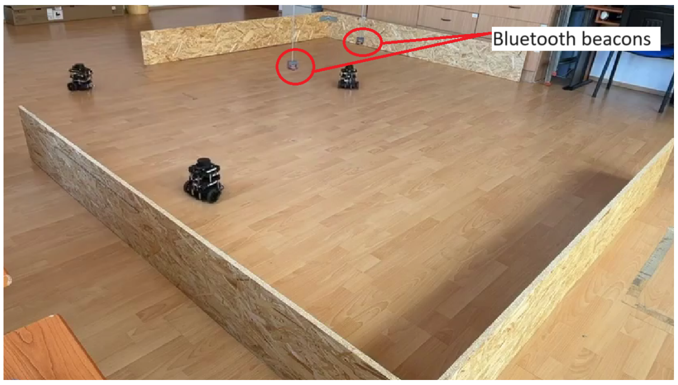


Figure 10: Overview of the TurtleBot3 experimental setup. The beacons are suspended on metal rods from the ceiling so that the robots can pass unhindered underneath. The plywood edges help with LiDAR-based localization.

of mobile robots. The antennas are arbitrarily placed indoors within an enclosed space. Figure 10 gives an overview of the experimental testbed.

For the antennas, two Infineon PSoC™ 4 development boards were setup as Bluetooth Low-Energy beacons, which constantly broadcast. Three ROBOTIS Turtlebot3 robots were used in the search. The Turtlebot3 is a unicycle mobile platform equipped with all of the necessary sensors and modules for self localization including inertial measurement unit, Light Detection And Ranging (LiDAR) and communication devices including WiFi and Bluetooth. The robots will seek the maxima of the RSSI field generated by the beacons. As the RSSI is a measure of an antenna's signal strength, these maxima will appear in close proximity to the sources.

Robot movement is bounded to a $[0.5, 3.5] \times [0.5, 3.5]\text{m}^2$ search space, inside of a $4 \times 4\text{m}^2$ safety perimeter partially surrounded by walls. The starting positions of the robots were chosen arbitrarily such that the robots are distributed over the search space: $x_0^1 = [0.7, 0.7]$, $x_0^2 = [2.5, 0.7]$, and $x_0^3 = [3.3, 3.0]$. The approximate coordinates of the two RSSI maxima (which are approximations because they were obtained by sampling on a grid) are $[1.69, 1.79]$ and $= [1.17, 3.05]$, respectively. The first two robots were configured as exploitation robots, whilst the third one as an exploration robot. The exclusion distance $\sigma = 0.5$ m, and the maximal velocity $\bar{v} = 0.22$ m/s.

To integrate VSOO with real hardware, a centralized controller was implemented that sends the robots to the targets whilst avoiding inter-agent collisions. This was achieved using a central computer which communicates with all of the robots using the Robot Operating System (ROS) infrastructure over a WiFi network. For each robot, an instance of the ROS navigation stack was run together with a translation layer that allows interoperability with Matlab. A simplified block diagram of the architecture can be found in Figure 11, where all of the blocks and signals with a q superscript should be understood as being repeated for each robot q. The flowchart in Figure 12 depicts the system's control flow and the information exchange between the centralized controller and the robots.

The ROS navigation stack is a modular robotics framework

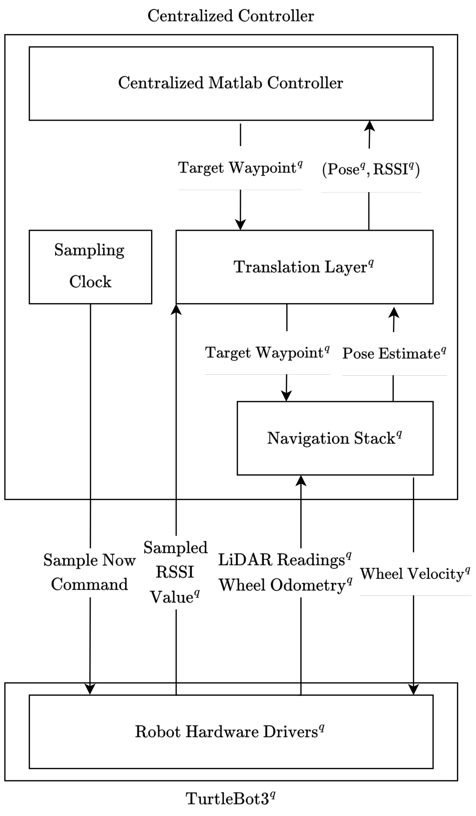
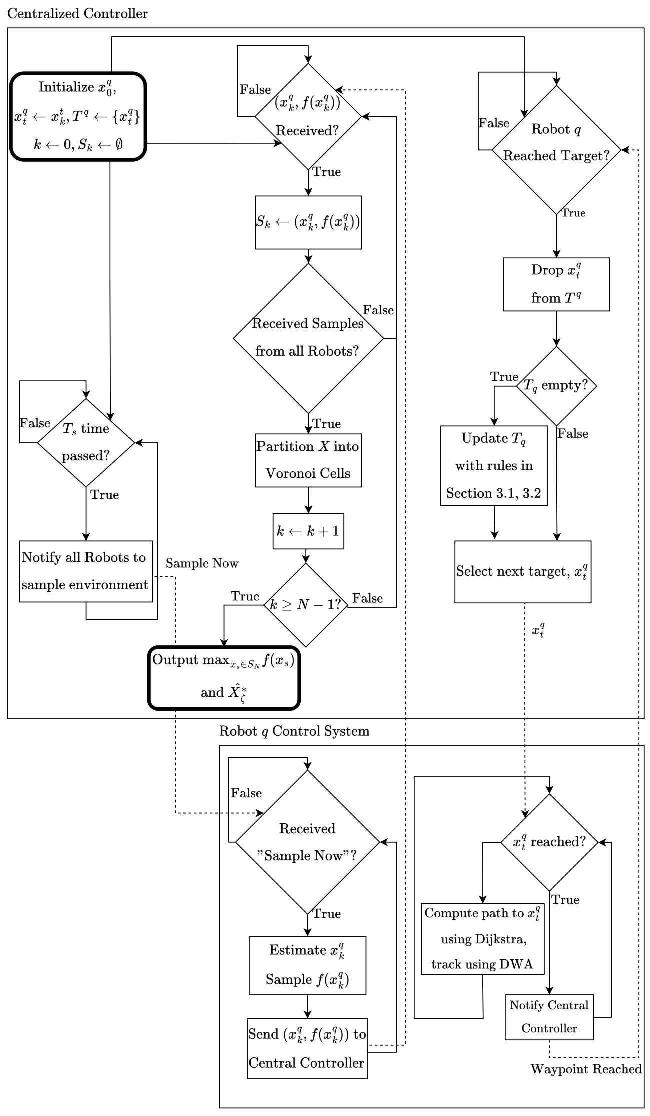


Figure 11: Simplified architecture diagram.

which integrates global and local planning with pose estimation to abstract the low-level control and localization for the user. The global planner is based on an extension of Dijkstra's algorithm (Dijkstra, 1959) and periodically recalculates a path for the robot to follow. A trajectory is generated by considering a discretized cost-map spanning the whole search space, where a higher cost is associated with an obstacle. The local planner employs the Dynamic Window Approach (Fox et al., 1997), which calculates the real-time control signals for the robot's wheel motors based on the required trajectory and obstacles detected. The pose estimator uses the adaptive Monte Carlo localization method (Dellaert et al., 1999) which fuses wheel odometry data and LiDAR readings. To achieve multi-robot planning, a plugin introduced by Chen (2020) was used, which allows for a centralized cost-map calculation based on the obstacles perceived using the LiDARs of all of the robots. In this configuration, each robot perceives the others as obstacles, around which a path will be generated.

To synchronize the sampling of the RSSI function between all of the robots, a ROS message is sent to all of the robots with a fixed frequency $f_s = 1$Hz. Whenever this message is received by a robot, the latest reading of the RSSI is sent to the interface node, which bundles the reading together with the corresponding pose estimate into a tuple. These tuples are forwarded to the Matlab centralized node.

Whenever a robot reaches its target, a callback message is sent to the Matlab node that triggers the calculation of a new waypoint using VSOO. The new goal is then sent to the navigation node for the robot to follow.



Centralized Controller
Initialize x_0^q, $x_t^q \leftarrow x_k^t, T^q \leftarrow \{x_t^q\}$ $k \leftarrow 0, S_k \leftarrow \emptyset$
False
$(x_k^q, f(x_k^q))$ Received?
True
$S_k \leftarrow (x_k^q, f(x_k^q))$
Received Samples from all Robots?
False
True
Partition X into Voronoi Cells
$k \leftarrow k+1$
$k \geq N-1$?
True
False
Output $\max_{x_s \in S_N} f(x_s)$ and $\hat{X}^*_\zeta$
False
T_s time passed?
True
Notify all Robots to sample environment
Sample Now
False
Robot q Reached Target?
True
Drop x_t^q from T^q
True
T_q empty?
False
Update T_q with rules in Section 3.1, 3.2
Select next target, x_t^q
x_t^q
Robot q Control System
False
Received "Sample Now"?
True
Estimate x_k^q Sample $f(x_k^q)$
Send $(x_k^q, f(x_k^q))$ to Central Controller
x_t^q reached?
True
Compute path to x_t^q using Dijkstra, track using DWA
Notify Central Controller
Waypoint Reached

Figure 12: Flowchart of the VSOO method. The centralized controller is split into three asynchronous processes (stemming from the initialization block at the top left) which wait for a specific event to trigger execution. Solid lines represent state transitions, while dotted lines represent messages passed over network between processes and devices. Program start and end blocks are circled with thicker lines.

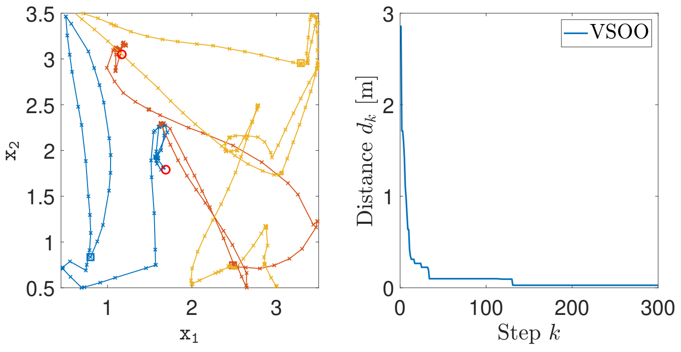


Figure 13: Trajectories of the robots in the real experiment (left). Performance of VSOO (right).

Figure 13 reports the experimental results for a representative run of the method. The left plot of the figure shows the robots' trajectories, while the right plot illustrates the performance of VSOO. The initial positions of the robots are marked with squares, the RSSI maxima locations with circles, and the 'x' symbols indicate sample locations. While robot 3 (yellow) continues to explore the search space, the exploitation robots 1 (blue) and 2 (red) converge to neighborhoods of the beacons, finding approximate maximal-RSSI locations at $[1.6, 1.92]$ and $[1.1, 3.13]$, both of them closer than 0.2m to their respective source.

The experiment in Figure 13 also exemplifies Theorem 4 and Remark 9 in practice. Specifically, 2 exploitation robots search for an equal number of 2 global optima distanced more than σ-away from each other. Moreover, the theoretical RSSI profile is expected to decay uniformly radially in a neighborhood of the transmitters (Atanasov et al., 2012; Zhu et al., 2013). Following Theorem 4, each such robot will eventually focus its search on a different x^*, reaching these optima at rather fast rates. This is also the case in our experiment, as seen in the left plot of Figure 13 and in the timelapse video of the experiment, available online at `http://rocon.utcluj.ro/files/vsoo_demo.mp4`.

7. Conclusion

This work introduced Voronoi Simultaneous Optimistic Optimization (VSOO), a novel divide-the-best-based method for mobile robots to find as quickly as possible all global optima of a Lipschitz-continuous objective function defined over their search area. We analytically concluded that VSOO (i) exhibits everywhere-dense and global convergence, and (ii) has well-characterized convergence rates for representative function shapes. Extensive numerical simulations on benchmark optimization functions, including GKLS-generated test functions, demonstrated that (iii) VSOO outperforms a series of representative global source/extremum seeking techniques, by approaching faster the global optima and always finding all these optima, with (iv) competitive execution time. Finally, (v) VSOO was validated in real-robot experiments where TurtleBot3 robots successfully identified the strongest wireless signals, effectively locating the antennas.

Future work will explore dynamically adapting the number of exploration/exploitation robots in VSOO. Moreover, we plan to extend VSOO to accommodate more complex robot dynamics, implement the method on diverse robotic platforms such as aerial drones and underwater robots, and apply it to more intricate real-world scenarios, including mapping and collection of surface or seabed litter.

Acknowledgments

This work was been financially supported from SeaClear2.0, a project that received funding from the European Climate, Infrastructure and Environment Executive Agency (CINEA) under grant agreement No 101093822; from DECIDE, no. 57/14.11.2022 funded under the PNRR I8 scheme by the Romanian Ministry of Research, Innovation, and Digitisation; and by a grant of the Ministry of Research, Innovation and Digitization, CNCS/CCCDI-UEFISCDI, project number PN-IV-P8-8.1-PRE-HE-ORG-2023-0012, within PNCDI IV.

References

Atanasov, N., Le Ny, J., Michael, N., Pappas, G. J., 2012. Stochastic source seeking in complex environments. In: 2012 IEEE International Conference on Robotics and Automation. IEEE, pp. 3013–3018.

Azuma, S., Sakar, M. S., Pappas, G. J., 2012. Stochastic source seeking by mobile robots. IEEE Transactions on Automatic Control 57 (9), 2308–2321.

Basiri, M., Schill, F., Lima, P. U., Floreano, D., 2012. Robust acoustic source localization of emergency signals from micro air vehicles. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 4737–4742.

Bayat, B., Crasta, N., Li, H., Ijspeert, A., 2016. Optimal search strategies for pollutant source localization. In: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Ieee, pp. 1801–1807.

Bourne, J. R., Pardyjak, E. R., Leang, K. K., 2019. Coordinated Bayesian-based bioinspired plume source term estimation and source seeking for mobile robots. IEEE Transactions on Robotics 35 (4), 967–986.

Boussaïd, I., Lepagnot, J., Siarry, P., 2013. A survey on optimization metaheuristics. Information sciences 237, 82–117.

Buşoniu, L., Varma, V. S., Lohéac, J., Codrean, A., Ştefan, O., Morărescu, I.-C., Lasaulce, S., 2020. Learning control for transmission and navigation with a mobile robot under unknown communication rates. Control Engineering Practice 100, 104460.

Castaneda, V., Mateus, D., Navab, N., 2011. SLAM combining ToF and high-resolution cameras. In: 2011 IEEE Workshop on Applications of Computer Vision (WACV). IEEE, pp. 672–678.

Chen, Y.-W., 2020. Multiple TurtleBot3 Navigation Environment.
URL `https://github.com/airuchen/multi_turtlebot3`

Conway, J. H., Sloane, N. J. A., 2013. Sphere packings, lattices and groups. Vol. 290. Springer Science & Business Media.

Dellaert, F., Fox, D., Burgard, W., Thrun, S., 1999. Monte Carlo localization for mobile robots. In: Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C). Vol. 2. pp. 1322–1328 vol.2.

Dijkstra, E. W., 1959. A note on two problems in connexion with graphs. Numerische Mathematik 1, 269–271.

Dorigo, M., Blum, C., 2005. Ant colony optimization theory: A survey. Theoretical computer science 344 (2-3), 243–278.

Fink, J., Kumar, V., 2010. Online methods for radio signal mapping with mobile robots. In: 2010 IEEE International Conference on Robotics and Automation. IEEE, pp. 1940–1945.

Floudas, C. A., Akrotirianakis, I. G., Caratzoulas, S., Meyer, C. A., Kallrath, J., 2005. Global optimization in the 21st century: Advances and challenges. Computers & Chemical Engineering 29 (6), 1185–1202.

Fox, D., Burgard, W., Thrun, S., 1997. The dynamic window approach to collision avoidance. IEEE Robotics & Automation Magazine 4 (1), 23–33.

Francis, A., Li, S., Griffiths, C., Sienz, J., 2022. Gas source localization and mapping with mobile robots: A review. Journal of Field Robotics 39 (8), 1341–1373.

Frazier, P. I., 2018. A tutorial on bayesian optimization.
URL `https://arxiv.org/abs/1807.02811`

Fu, L., Özgüner, Ü., 2011. Extremum seeking with sliding mode gradient estimation and asymptotic regulation for a class of nonlinear systems. Automatica 47 (12), 2595–2603.

Fu, Z., Chen, Y., Ding, Y., He, D., 2019. Pollution source localization based on multi-UAV cooperative communication. Ieee Access 7, 29304–29312.

Gaviano, M., Kvasov, D. E., Lera, D., Sergeyev, Y. D., 2003. Algorithm 829: Software for generation of classes of test functions with known local and global minima for global optimization. ACM Trans. Math. Softw. 29 (4), 469–480.
URL `https://doi.org/10.1145/962437.962444`

Ghadiri-Modarres, M., Mojiri, M., 2020. Normalized extremum seeking and its application to nonholonomic source localization. IEEE Transactions on automatic control 66 (5), 2281–2288.

Ghassemi, P., Balazon, M., Chowdhury, S., 2022. A penalized batch-bayesian approach to informative path planning for decentralized swarm robotic search. Autonomous Robots 46 (6), 725–747.

Graefenstein, J., Bouzouraa, M. E., 2008. Robust method for outdoor localization of a mobile robot using received signal strength in low power wireless networks. In: 2008 IEEE International Conference on Robotics and Automation. IEEE, pp. 33–38.

Grisetti, G., Kümmerle, R., Stachniss, C., Burgard, W., 2010. A tutorial on graph-based SLAM. IEEE Intelligent Transportation Systems Magazine 2 (4), 31–43.

Gronemeyer, M., Alpen, M., Horn, J., 2020. Limited gradient criterion for global source seeking with mobile robots. IFAC-PapersOnLine 53 (2), 15288–15293.

Gronemeyer, M., Bartels, M., Werner, H., Horn, J., 2017. Using particle swarm optimization for source seeking in multi-agent systems. IFAC-PapersOnLine 50 (1), 11427–11433.

Hales, T. C., 2011. Sphere packings, i. The Kepler Conjecture: The Hales-Ferguson Proof, 379–431.

He, X., Steiner, J. A., Bourne, J. R., Leang, K. K., 2019. Gaussian-based kernel for multi-agent aerial chemical-plume mapping. In: Dynamic Systems and Control Conference. Vol. 59162. American Society of Mechanical Engineers, p. V003T21A004.

Hess, W., Kohler, D., Rapp, H., Andor, D., 2016. Real-time loop closure in 2D LIDAR SLAM. In: 2016 IEEE international conference on robotics and automation (ICRA). IEEE, pp. 1271–1278.

Hoshiba, K., Washizaki, K., Wakabayashi, M., Ishiki, T., Kumon, M., Bando, Y., Gabriel, D., Nakadai, K., Okuno, H. G., 2017. Design of UAV-embedded microphone array system for sound source localization in outdoor environments. Sensors 17 (11), 2535.

Jing, T., Meng, Q.-H., Ishida, H., 2021. Recent progress and trend of robot odor source localization. IEEJ Transactions on Electrical and Electronic Engineering 16 (7), 938–953.

Jones, D., Perttunen, C., Stuckman, B., 1993. Lipschitzian optimization without the Lipschitz constant. Journal of Optimization Theory and Applications 79 (1), 157 – 181.

Kennedy, J., Eberhart, R., Shi, Y., 2001. Swarm Intelligence. Evolutionary Computation Series. Elsevier Science.

Khong, S. Z., Manzie, C., Tan, Y., Nešić, D., 2014a. Multi-agent gradient climbing via extremum seeking control. IFAC Proceedings Volumes 47 (3), 9973–9978.

Khong, S. Z., Tan, Y., Manzie, C., Nešić, D., 2014b. Multi-agent source seeking via discrete-time extremum seeking control. Automatica 50 (9), 2312–2320.

Kim, D., 2023. Engineered Underwater Vehicle for Ocean Litter Mapping. New Mexico Journal of Science 57.

Kivelevitch, E., 2024. MDMTSPV-GA - Multiple Depot Multiple Traveling Salesmen Problem solved by Genetic Algorithm. Retrieved June 6, 2024, from MATLAB Central File Exchange.

Krstić, M., Wang, H.-H., 2000. Stability of extremum seeking feedback for general nonlinear dynamic systems. Automatica 36 (4), 595–601.

Kvasov, D. E., Sergeyev, Y. D., 2015. Deterministic approaches for solving practical black-box global optimization problems. Advances in Engineering Software 80, 58–66.

Lalish, E., Morgansen, K. A., 2008. Decentralized reactive collision avoidance for multivehicle systems. In: 2008 47th IEEE Conference on Decision and Control. IEEE, pp. 1218–1224.

Lera, D., Nasso, M. C., Posypkin, M., Sergeyev, Y. D., 2024. Determining solution set of nonlinear inequalities using space-filling curves for finding working spaces of planar robots. Journal of Global Optimization 89 (2), 415–434.

Lera, D., Posypkin, M., Sergeyev, Y. D., 2021. Space-filling curves for numerical approximation and visualization of solutions to systems of nonlinear inequalities with applications in robotics. Applied Mathematics and Computation 390, 125660.

Li, Y., Liu, T., Zhou, E., Zhang, F., 2021. Bayesian learning model predictive control for process-aware source seeking. IEEE Control Systems Letters 6, 692–697.

Liu, S.-J., Krstic, M., 2010. Stochastic averaging in continuous time and its applications to extremum seeking. IEEE Transactions on Automatic Control 55 (10), 2235–2250.

Lizotte, D. J., 2008. Practical bayesian optimization. PhD Thesis, Department of Computing Science, University of Alberta.

Matveev, A. S., Hoy, M. C., Savkin, A. V., 2014. 3D environmental extremum seeking navigation of a nonholonomic mobile robot. Automatica 50 (7), 1802–1815.

Mockus, J., 2005. The bayesian approach to global optimization. In: System Modeling and Optimization: Proceedings of the 10th IFIP Conference New York City, USA, August 31–September 4, 1981. Springer, pp. 473–481.

Munos, R., 2011. Optimistic optimization of a deterministic function without the knowledge of its smoothness. Advances in neural information processing systems 24.

Munos, R., 2014. From bandits to Monte-Carlo Tree Search: The optimistic principle applied to optimization and planning. Foundations and Trends in Machine Learning 7 (1), 1–129.

Nguyen, H. G., Pezeshkian, N., Raymond, M., Gupta, A., Spector, J. M., 2003. Autonomous communication relays for tactical robots. In: Proceedings of the International Conference on Advanced Robotics (ICAR). pp. 35–40.

Paulavičius, R., Žilinskas, J., 2014. Simplicial partitions in global optimization. Springer.

Pintér, J. D., 1995. Global optimization in action: continuous and Lipschitz optimization: algorithms, implementations and applications. Vol. 6. Springer Science & Business Media.

Plaku, E., Kavraki, L. E., Vardi, M. Y., 2010. Motion planning with dynamics by a synergistic combination of layers of planning. IEEE Transactions on Robotics 26 (3), 469–482.

Poveda, J. I., Krstić, M., 2020. Fixed-time gradient-based extremum seeking. In: 2020 American Control Conference (ACC). IEEE, pp. 2838–2843.

Ramirez-Llanos, E., Martinez, S., 2018. Stochastic source seeking for mobile robots in obstacle environments via the SPSA method. IEEE Transactions on Automatic Control 64 (4), 1732–1739.

Saitou, T., Nukada, M., Uchimura, Y., 2013. Deployment control of mobile robots for wireless network relay based on received signal strength. In: 2013 IEEE Workshop on Advanced Robotics and its Social Impacts. IEEE, pp. 237–242.

Sântejudean, T., Buşoniu, L., 2022. Online learning control for path-aware global optimization with nonlinear mobile robots. Control Engineering Practice 126, 105228.

Sântejudean, T., Ungur, Ş., Herzal, R., Morărescu, I.-C., Varma, V. S., Buşoniu, L., 2025. Globally convergent path-aware optimization with mobile robots. Nonlinear Analysis: Hybrid Systems 55, 101546.

Scheinker, A., 2024. 100 years of extremum seeking: A survey. Automatica 161, 111481.

Scheinker, A., Krstić, M., 2017. Model-free stabilization by extremum seeking. Springer.

Schmid, L., Pantic, M., Khanna, R., Ott, L., Siegwart, R., Nieto, J., 2020. An efficient sampling-based method for online informative path planning in unknown environments. IEEE Robotics and Automation Letters 5 (2), 1500–1507.

Sergeyev, Y. D., 1998. On convergence of" divide the best" global optimization algorithms. Optimization 44 (3), 303–325.

Sergeyev, Y. D., Kvasov, D., Mukhametzhanov, M., 2018. On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. Scientific reports 8 (1), 453.

Sergeyev, Y. D., Kvasov, D. E., 2006. Global search based on efficient diagonal partitions and a set of lipschitz constants. SIAM Journal on Optimization 16 (3), 910–937.

Sergeyev, Y. D., Kvasov, D. E., 2015. A deterministic global optimization using smooth diagonal auxiliary functions. Communications in Nonlinear Science and Numerical Simulation 21 (1-3), 99–111.

Sergeyev, Y. D., Kvasov, D. E., 2017. Deterministic global optimization: An introduction to the diagonal approach. Springer.

Sergeyev, Y. D., Nasso, M. C., Lera, D., 2024. Numerical methods using two different approximations of space-filling curves for black-box global optimization. Journal of Global Optimization 88 (3), 707–722.

Sergeyev, Y. D., Strongin, R. G., Lera, D., 2013. Introduction to global optimization exploiting space-filling curves. Springer Science & Business Media.

Sibanyoni, S. V., Ramotsoela, D. T., Silva, B. J., Hancke, G. P., 2018. A 2-d acoustic source localization system for drones in search and rescue missions. IEEE Sensors Journal 19 (1), 332–341.

Šiljeg, A., Marić, I., Krekman, S., Cukrov, N., Lovrić, M., Domazetović, F., Panda, L., Bulat, T., 2023. Mapping of marine litter on the seafloor using WASSP S3 multibeam echo sounder and Chasing M2 ROV. Frontiers in Earth Science 11, 1133751.

Snoek, J., Larochelle, H., Adams, R. P., 2012. Practical bayesian optimization of machine learning algorithms.
URL `https://arxiv.org/abs/1206.2944`

Spears, D., Zarzhitsky, D., Thayer, D., 2005. Multi-robot chemical plume tracing. In: Multi-Robot Systems. From Swarms to Intelligent Automata Volume III: Proceedings from the 2005 International Workshop on Multi-Robot Systems. Springer, pp. 211–221.

Stripinis, L., Paulavičius, R., 2023. Derivative-Free DIRECT-Type Global Optimization: Applications and Software. Springer Nature.

Stripinis, L., Paulavičius, R., 2024. Lipschitz-inspired halrect algorithm for derivative-free global optimization. Journal of Global Optimization 88 (1), 139–169.

Strongin, R. G., 1978. Numerical methods in multiextremal problems. Moscow, USSR: Nauka.

Strongin, R. G., Sergeyev, Y. D., 2013. Global optimization with non-convex constraints: Sequential and parallel algorithms. Vol. 45. Springer Science & Business Media.

Surjanovic, S., Bingham, D., 2013. Virtual Library of Simulation Experiments: Test Functions and Datasets. Retrieved April 17, 2024.
URL `http://www.sfu.ca/~ssurjano`

Suttner, R., Krstic, M., 2023. Nonlocal Nonholonomic Source Seeking Despite Local Extrema. arXiv preprint arXiv:2303.16027.

Taketomi, T., Uchiyama, H., Ikeda, S., 2017. Visual SLAM algorithms: A survey from 2010 to 2016. IPSJ transactions on computer vision and applications 9, 1–11.

Tan, C. S., Mohd-Mokhtar, R., Arshad, M. R., 2021. A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms. IEEE Access 9, 119310–119342.

Tan, Y., Moase, W. H., Manzie, C., Nešić, D., Mareels, I. M., 2010. Extremum seeking from 1922 to 2010. In: Proceedings of the 29th Chinese control conference. IEEE, pp. 14–26.

Tomar, V., Bansal, M., Singh, P., 2024. Metaheuristic algorithms for optimization: A brief review. Engineering Proceedings 59 (1), 238.

Vasilijević, A., Nad, D., Mandić, F., Mišković, N., Vukić, Z., 2017. Coordinated navigation of surface and underwater marine robotic vehicles for ocean sampling and environmental monitoring. IEEE/ASME transactions on mechatronics 22 (3), 1174–1184.

Veettil, B. K., Quan, N. H., Hauser, L. T., Van, D. D., Quang, N. X., 2022. Coastal and marine plastic litter monitoring using remote sensing: A review. Estuarine, Coastal and Shelf Science 279, 108160.

Wang, Q., Debbarma, D., Lo, A., Cao, Z., Niemegeers, I., Heemstra de Groot, S., 05 2015. Distributed Antenna System for Mitigating Shadowing Effect in 60GHz WLAN. Wireless Personal Communications 82.

Wang, Z., Zhou, X., Wang, J., 2022. Extremum-seeking-based adaptive model-free control and its application to automated vehicle path tracking. IEEE/ASME Transactions on Mechatronics 27 (5), 3874–3884.

Xu, X., Zhang, L., Yang, J., Cao, C., Wang, W., Ran, Y., Tan, Z., Luo, M., 2022. A review of multi-sensor fusion slam systems based on 3D LIDAR. Remote Sensing 14 (12), 2835.

Yılmaz, A., Yilmaz, F., Alouini, M.-S., Kucur, O., 01 2013. On the Performance of Transmit Antenna Selection Based on Shadowing SideInformation. IEEE Transactions on Vehicular Technology 62, 454–460.

Zhang, C., Ordóñez, R., 2011. Extremum-seeking control and applications: a numerical optimization-based approach. Springer Science & Business Media.

Zhang, L., Yang, J., Yu, X., 2023. Disturbance rejection extremum seeking: A sliding mode control approach. IEEE Control Systems Letters 7, 1447–1452.

Zhigljavsky, A., Žilinskas, A., 2021. Bayesian and high-dimensional global optimization. Springer.

Zhu, S., Wang, D., Low, C. B., 2013. Cooperative control of multiple UAVs for moving source seeking. In: 2013 International Conference on Unmanned Aircraft Systems (ICUAS). IEEE, pp. 193–202.

Zhu, S., Wang, D., Low, C. B., 2014. Cooperative control of multiple UAVs for moving source seeking. Journal of Intelligent & Robotic Systems 74, 333–346.

Zou, R., Kalivarapu, V., Winer, E., Oliver, J., Bhattacharya, S., 2015. Particle swarm optimization-based source seeking. IEEE Transactions on Automation Science and Engineering 12 (3), 865–875.