Error-Bounded Approximation of Pareto Fronts in Robot Planning Problems

Alexander Botros[*], Armin Sadeghi[*], Nils Wilde[†], Javier Alonso-Mora[†], and Stephen L. Smith[*]

[*] Department of Electrical and Computer Engineering
University of Waterloo, Waterloo, ON, Canada
{alexander.botros, a6sadegh, stephen.smith}@uwaterloo.ca,
[†] Department for Cognitive Robotics, 3ME
Delft University of Technology, Delft, Netherlands
{n.wilde, j.alonsomora}@tudelft.nl,

Abstract. Many problems in robotics seek to simultaneously optimize several competing objectives under constraints. A conventional approach to solving such multi-objective optimization problems is to create a single cost function comprised of the weighted sum of the individual objectives. Solutions to this scalarized optimization problem are Pareto optimal solutions to the original multi-objective problem. However, finding an accurate representation of a Pareto front remains an important challenge. Using uniformly spaced weight vectors is often inefficient and does not provide error bounds. Thus, we address the problem of computing a finite set of weight vectors such that for any other weight vector, there exists an element in the set whose error compared to optimal is minimized. To this end, we prove fundamental properties of the optimal cost as a function of the weight vector, including its continuity and concavity. Using these, we propose an algorithm that greedily adds the weight vector least-represented by the current set, and provide bounds on the error. Finally, we illustrate that the proposed approach significantly outperforms uniformly distributed weights for different robot planning problems with varying numbers of objective functions.

Keywords: Multi-objective optimization, Planning, Human-robot interaction

1 Introduction

Robot planning problems often face the challenge of simultaneously optimizing multiple objectives. Finding the appropriate trade-off between objectives remains a major challenge when deploying intelligent autonomous systems. For instance,

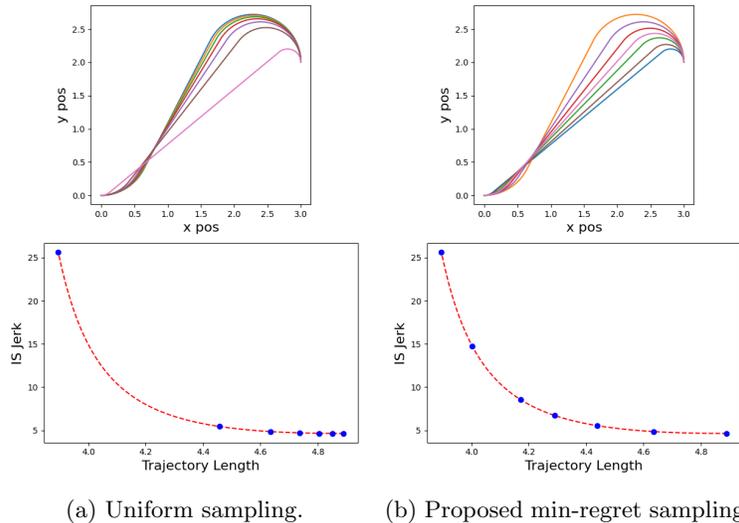(a) Uniform sampling.   (b) Proposed min-regret sampling.

Fig. 1: *Top:* Different optimal trade-offs between trajectory length and discomfort, sampled uniformly and by the proposed method. *Bottom:* Approximated Pareto front (blue) and ground truth Pareto front (red).

autonomous cars optimize trajectories for passenger comfort and efficiency [1,2], manipulators working in cooperative tasks consider human ergonomics as well as risk of collision or other damage in handover tasks [3], mobile robots performing transportation in industrial settings trade-off task efficiency and compliance with user specific norms [4], and autonomous mobility-on-demand systems seek to maximize service quality while minimizing operation cost [5].

A common technique in multi-objective optimization is to create a single cost function defined by the weighted sum of the individual objectives [6]. This approach, called *linear scalarization*, leads to solutions that are Pareto optimal for the multi-objective problem [7]. That is, the solution to the scalarized single-objective problem cannot be changed to improve the value of one of its objectives without degrading the value of another. This work is motivated by two classes of problems in robotics: 1) obtaining near optimal solutions to a linearly scalarized multi-objective optimization problem (LSMOP) for any given weight vectors, and 2) learning the preferred solution behavior of a human user. Applications of the first class include [5] in which an adjustable trade-off between the service quality and operating costs for autonomous mobility-on-demand systems are optimized. Conversely, the second class of problems seeks to compute a weight vector representing the relative importance of each objective to a user given some knowledge of that users' preferred solutions [4,8–12].

In either class, it is often beneficial to pre-compute solutions to the LSMOP for a set of weight vectors. If the LSMOP is computationally intensive to solve, requiring online solutions may be impractical. This motivates the problem of finding a set of weights and their corresponding optimal solutions such that for

any possible weight vector, there exists an element of the set whose solution is close to optimal. A naive approach would involve densely sampling the set of all possible weight vectors. However, the sensitivity of some objectives to changes in solution may result in a skewed sampling of Pareto optimal solutions. This is illustrated in Figure 1 where we seek to compute a Dubins trajectory between fixed start and goal configurations that minimizes a trade-off between trajectory length and discomfort (measured as the integral of the squared jerk over the trajectory [13]). In the left images, a set of weights vectors is selected uniformly and the resulting solution trajectories (top) and Pareto front (bottom) is shown. We observe that uniformly sampling over the set of possible weights generate a set of similar trajectories. In this paper, we propose a technique to construct a set of weight vectors such that their corresponding optimal solutions provide homogeneous coverage of the Pareto front (see Figure 1b).

Contributions: The contributions of this work are fourfold: First, we propose an algorithm for homogeneous sampling of the Pareto front. Second, we prove fundamental properties of the optimal cost for any weight such as concavity and continuity. Third, we show that using the best available solution from the computed set to approximate the optimal solution to LSMOP for any arbitrary weight has has a bounded error. Finally, we showcase the advantages of our sampling algorithm in different robotics applications, namely trajectory planning and reward learning.

1.1   Related Work

Linear Scalarization in Robotics: The simplicity of linear scalarization has made it one of the most widely used tools in robotic multi-objective optimization [14]. Though the technique is not able to capture all Pareto-optimal solutions for non-convex fronts, it does guarantee that all LSMOP solutions are Pareto-optimal. In [15], a trajectory smoothing algorithm is proposed based on the weighted sum of the competing objectives: trajectory length, smoothness, and obstacle distances. The authors of [16] minimize the weighted trade-off between mission completion time and communication outage duration in the navigation of cellular-connected UAVs, while in [17], linear scalarization is used to optimize robotic limitations and observation rewards for use in autonomous human activity tracking.

In human-robot interation (HRI), weight vectors are used to represent the relative importance of objectives – often referred to as features – to a user [8, 11, 12, 18]. In *reward learning* it is the objective to learn a user's weight vector using interactions such as demonstrations, corrections, or choice feedback. In order to expedite the learning process, feasible solutions for the multi-objective optimization problem are often pre-computed and shown to the user who then provides feedback. In [8, 12, 19], each pre-computed solution is generated with random action sequences. Thus, the solutions used in the learning process are usually not optimal for any weight. In [11, 20], the authors pre-compute Pareto-optimal solutions enabling an active learning method based on regret leading to significant improvement over randomly generated solutions.

However, the work in [11, 20] rely on uniformly sampled weight vectors. Though this approach asymptotically covers the set of all LSMOP-optimal solutions, it can be inefficient as different weight vectors can have very similar, or even identical solutions. The authors of [21] observed that the presence of similar solution strongly influences the Boltzmann decision model which is commonly used in HRI. They propose a decision model where a similarity metric corrects the bias induced by a high number of similar trajectories. Similarly, in our work we are interested in finding solutions with dissimilar features. While [21] handles the over-representation of similar solutions in the ground set with their proposed decision model, we instead address the problem at an earlier stage: our algorithm can be used to generate a ground set where similarities are minimized.

Uniform Sampling of Pareto-Optimal Solutions: Uniform sampling of a Pareto front is a persistent problem in multi-objective optimization. Similar to our work, the authors of [22] offer a technique of Pareto-uniform sampling based on equispacing constraints. However, this work only considers the case of two competing objectives. Further, they accomplish their goal by solving a significantly harder problem than the original LSMOP. Also similar, the work in [23] proposes a set of weight vectors that approximately uniformly cover a Pareto front specifically for use in the design of robots. The authors design the set that minimizes the total squared error between the value of the objectives in the set and heuristic objectives. It therefore relies on the approximate optimality of these objectives. The work in [24] proposes a method to cover the set of Pareto-optimal solutions specifically for use in reinforcement learning applications. There, the authors seek to compute policies that maximize expected returns by computing, storing, and updating a set of samples. In [14] the authors provide a means of exploring a (possibly non-convex) Pareto front in order to obtain a solution that is near-optimal for a user. That work starts with an initial guess solution and then moves in a direction according to the preferences of a user. While the convexity of the Pareto front (a requirement for linear scalarization to obtain all Pareto-optimal solutions) is not assumed, their technique does require solving the LSMOP online as the Pareto front is explored. Moreover, the requirement of a user-preferred direction is not assumed in our work. In [25] a Pareto front approximation is proposed using Markov chain random walks. Their goal is to *uniformly* place samples on the Pareto front, while our goal is to *minimize error* in the space of Pareto-optimal costs.

2   Problem Statement

For $n \in \mathbb{N}$, a general multi-objective optimization problem (MOP) is of the form

$$\min_{s \in S} \left(f_1(s), f_2(s), \dots, f_n(s) \right).$$

Here, the set of feasible solutions given constraints is denoted S, and it is desired to simultaneously minimize n objectives $f_i(s), i = 1, \dots, n$. The linear scalarization of the MOP above would involve the creation of a single cost function by

introducing a vector of weights $\boldsymbol{w} = [w_1, w_2, \ldots, w_n] \in \mathbb{R}^n_{\geq 0}$. Let $c(s, \boldsymbol{w})$ denote the cost of the solution s evaluated by the weights $\boldsymbol{w}$, i.e.,

$$c(s, \boldsymbol{w}) = \sum_{i=1}^{n} w_i \, f_i(s) = \boldsymbol{w}^\top \boldsymbol{f}(s),$$

where $\boldsymbol{f}(s) = [f_1(s), \ldots, f_n(s)]$, $\forall s \in S$. The resulting linearly scalarized multi-objective optimization problem (LSMOP) is to solve

$$u(\boldsymbol{w}) = \min_{s \in S} c(s, \boldsymbol{w}). \tag{1}$$

For any weight $\boldsymbol{w} \in \mathbb{R}^n_{\geq 0}$, solution $s \in S$, and $\lambda \in \mathbb{R}_{>0}$, it holds that $c(s, \lambda \boldsymbol{w}) = \lambda c(s, \boldsymbol{w})$ implying that a minimizer of $c(s, \boldsymbol{w})$ also minimizes $c(s, \lambda \boldsymbol{w})$. Further, if $\boldsymbol{w} = [0, 0, \ldots, 0]$, then $u(\boldsymbol{w})$ is trivially 0. Thus, given $\boldsymbol{w} = [w_1, \ldots, w_n]$ where not all elements are identically 0, and letting $\lambda = \left(\sum_{i=1}^{n} w_i\right)^{-1}$, we can obtain all non-trivial optimal solutions $u(\boldsymbol{w})$ for all $\boldsymbol{w} \in \mathbb{R}^n_{\geq 0}$ using weights $\boldsymbol{w} \in W$ where

$$W = \left\{ \boldsymbol{w} \in \mathbb{R}^n_{\geq 0}, \sum_{i=1}^{n} w_i = 1 \right\}. \tag{2}$$

We refer to the set W as the *weight space*, and we use the notation

$$s^*(\boldsymbol{w}) = \arg\min_{s \in S} \boldsymbol{w}^\top \boldsymbol{f}(s), \; \forall \boldsymbol{w} \in W,$$

implying that $u(\boldsymbol{w}) = c(s^*(\boldsymbol{w}), \boldsymbol{w})$ by (1). We make the following assumptions:

Assumption 1 (Exact Solution)**.** An exact solver exists for the optimization problem (1).

Assumption 2 (Bounded Objectives)**.** For any weight $\boldsymbol{w} \in \mathbb{R}^n_{\geq 0}$, and any solution $s^*(\boldsymbol{w})$, the objectives $\boldsymbol{f}(s^*(\boldsymbol{w}))$ are bounded.

In this work, we propose a method to compute a finite set of weights $\Omega \subseteq W$ that will, for any $\boldsymbol{w}^* \in W$, allow us to approximate $u(\boldsymbol{w}^*)$ with $u(\boldsymbol{w}')$ for an appropriately chosen $\boldsymbol{w}' \in \Omega$. To evaluate the quality of a candidate set Ω, we use the notion of regret from [11, 26], defined formally here:

Definition 1 (Regret). *Given two weights $\boldsymbol{w}', \boldsymbol{w}^* \in W$, the regret of $\boldsymbol{w}'$ under $\boldsymbol{w}^*$ is defined as*

$$r(\boldsymbol{w}' | \boldsymbol{w}^*) = \boldsymbol{w}^{*\top} \boldsymbol{f}(s^*(\boldsymbol{w}')) - u(\boldsymbol{w}^*). \tag{3}$$

Intuitively, $r(\boldsymbol{w}' | \boldsymbol{w}^*)$ represents the *error* in cost incurred by using an optimal solution given weight $\boldsymbol{w}'$ (given by $s^*(\boldsymbol{w}')$) to approximate a solution given weight $\boldsymbol{w}^*$. We now formally state the main problem addressed in this work.

Problem 1 (Min-Max Regret Sampling). For the LSMOP (1) and some integer $K > 0$, find a set of sampled weights Ω that solves

$$\min_{\boldsymbol{w} \in W} \max_{\boldsymbol{w}' \in \Omega} \min r(\boldsymbol{w}' | \boldsymbol{w}^*)$$
$$s.t. \; |\Omega| \leq K. \tag{4}$$

Given a weight $\boldsymbol{w} \in \mathcal{W}$ and a set $\Omega \subseteq \mathcal{W}$, the first minimization in (4), $\min_{\boldsymbol{w}' \in \Omega} r(\boldsymbol{w}'|\boldsymbol{w})$ represents the sub-optimality of approximating a solution $s^*(\boldsymbol{w})$ with a solution $s^*(\boldsymbol{w}')$ where $\boldsymbol{w}'$ is the weight in Ω that minimizes this sub-optimality – i.e., $\boldsymbol{w}'$ is a best representative of $\boldsymbol{w}$ in Ω. The maximization $\max_{\boldsymbol{w} \in \mathcal{W}} \min_{\boldsymbol{w}' \in \Omega} r(\boldsymbol{w}'|\boldsymbol{w})$, represents the regret of the worst represented weight $\boldsymbol{w} \in \mathcal{W}$ by elements in Ω. We refer to the solution of this maximization as the *maximum regret given* Ω. In total, (4) seeks a set Ω such that the regret of the worst represented element in $\mathcal{W}$ is minimized.

In this paper, we offer an approximate solution to the optimization in (4) by way of an algorithm that computes a feasible solution Ω such that the maximum regret given Ω is bounded. In the next section, we provide the theoretical groundwork that makes this solution possible.

3   Problem Analysis

We begin with a structural analysis of the cost function $u(\boldsymbol{w})$ to derive an efficient algorithm for solving Problem 1. First, we make two critical observations.

Observation 1. Given any two weights $\boldsymbol{w}, \boldsymbol{w}' \in \mathcal{W}$, we have

$$u(\boldsymbol{w}) \leq \boldsymbol{w} \cdot \boldsymbol{f}(s^*(\boldsymbol{w}')), \tag{5}$$

That is an optimal solution given weights $\boldsymbol{w}$ will incur no higher cost than a solution that is optimal for some different weight vector $\boldsymbol{w}'$. Here, $s^*(\boldsymbol{w}')$ is optimal given weights $\boldsymbol{w}'$ but not necessarily optimal given weights $\boldsymbol{w}$. By (3), the inequality in (5) implies that $r(\boldsymbol{w}'|\boldsymbol{w}) \geq 0$.

Observation 2 (Optimal Cost Concavity)**.** The optimal cost function $u(\boldsymbol{w})$ is a concave function of $\boldsymbol{w}$. Indeed, for each $s \in S$, the cost $c(s, \boldsymbol{w}) = \boldsymbol{w} \cdot \boldsymbol{f}(s)$ is an affine function of $\boldsymbol{w}$ (and is therefore concave). Therefore, $u(\boldsymbol{w}) = \min_{s \in S} c(s, \boldsymbol{w})$ is concave [27][Section 3.2.3].

Observation 2 motivates the following Corollary:

Corollary 1 (Optimal Cost Continuity). *Under Assumptions 1, 2, the function* $u(\boldsymbol{w})$ *is continuous on the interior of* $\mathcal{W}$. *Further,* $u(\boldsymbol{w})$ *is continuous on the boundary of* $\mathcal{W}$ *in the direction of its interior.*

Proof. By Observation 2, $u(\boldsymbol{w})$ is concave in $\mathcal{W}$. Noting in addition that $\mathcal{W} \subseteq \mathbb{R}^n$ is convex, it must hold that $u(\boldsymbol{w})$ is continuous on the interior of $\mathcal{W}$. This is because concave functions are continuous on the interior of convex sets. Next, consider two weights $\boldsymbol{w}', \boldsymbol{w}'' \in \mathcal{W}$ one of which lies on the boundary of $\mathcal{W}$ and one in the interior of $\mathcal{W}$. Suppose that $||\boldsymbol{w}' - \boldsymbol{w}''|| \leq \delta$ for some $\delta > 0$ arbitrarily small, and – without loss of generality – that $u(\boldsymbol{w}') > u(\boldsymbol{w}'')$. Because it is assumed that $u(\boldsymbol{w})$ is bounded on $\mathcal{W}$, if it experiences a discontinuity on the line connecting $\boldsymbol{w}', \boldsymbol{w}''$, it must hold that $u(\boldsymbol{w})$ experiences a jump between $\boldsymbol{w}', \boldsymbol{w}''$. That is, there

must exist a $M \in \mathbb{R}_{>0}$ independent of δ such that $u(\mathbf{w}'') + M \leq u(\mathbf{w}')$. Since $||\mathbf{w}'' - \mathbf{w}'|| \leq \delta$, and $f(s\;(\mathbf{w}''))$ is bounded by Assumption 2, there must exist a value of δ sufficiently small so as to guarantee that $(\mathbf{w}' - \mathbf{w}'') \cdot f(s\;(\mathbf{w}'')) < M/2$. Therefore, by construction,

$$\mathbf{w}' \cdot f(s\;(\mathbf{w}'')) < \frac{M}{2} + \mathbf{w}'' \cdot f(s\;(\mathbf{w}'')) = \frac{M}{2} + u(\mathbf{w}'')$$

$$\leq \frac{M}{2} + u(\mathbf{w}') - M = u(\mathbf{w}') - \frac{M}{2} < u(\mathbf{w}').$$

Therefore, $r(\mathbf{w}''|\mathbf{w}') = \mathbf{w}' \cdot f(s(\mathbf{w}'')) - u(\mathbf{w}') < 0$ which is a contradiction by Observation 1. □

Critically, the previous results do not require unique solutions $s\;(\mathbf{w})$ or continuous objectives $f(s\;(\mathbf{w}))$. Extending these results:

Theorem 1 (Convexity of Regret). *For a fixed weight $\mathbf{w}' \in W$, the regret $r(\mathbf{w}'|\mathbf{w})$ is a convex function of $\mathbf{w}$.*

Proof. By the definition of regret in (3), $r(\mathbf{w}'|\mathbf{w}) = \mathbf{w} \cdot f(s\;(\mathbf{w}')) - u(\mathbf{w})$. where $\mathbf{w} \cdot f(s\;(\mathbf{w}'))$ is linear in $\mathbf{w}$ and $u(\mathbf{w})$ is concave by Observation 2. Thus, $r(\mathbf{w}'|\mathbf{w})$ is the difference of a linear and concave function of $\mathbf{w}$ which is convex. □

Because $u(\mathbf{w})$ is continuous and concave, it must hold that the function lies below any sub-gradient. This motivates the following Corollary which follows directly from the definition of $u(\mathbf{w})$ and the concavity of $c(s, \mathbf{w})$ for each fixed $s \in S$ [27][Section 6.5.5].

Corollary 2 (Sub-gradient Optimal Cost). *For any $\mathbf{w} \in W$, the sub-gradients of $u(\mathbf{w})$ are given by*

$$\partial u(\mathbf{w}) = f(s\;(\mathbf{w})),$$

for any minimizing solution $s\;(\mathbf{w}) \in S$.

Observe that sub-gradients are defined even for non-differentiable continuous functions. Further, The results above imply that given two weights $\mathbf{w}', \mathbf{w} \in W$, the regret $r(\mathbf{w}'|\mathbf{w})$ — which coincides with the error incurred by approximating a solution $s\;(\mathbf{w})$ with the solution $s\;(\mathbf{w}')$ — is exactly the error of approximating a concave function via *linear interpolation*. Indeed, the first order approximation of $u(\mathbf{w})$ given $u(\mathbf{w}')$ is given by $u(\mathbf{w}) \approx u(\mathbf{w}') + \nabla u(\mathbf{w}') \cdot (\mathbf{w} - \mathbf{w}')$ assuming u is differentiable at $\mathbf{w}'$. However, by Corollary 2, $\nabla u(\mathbf{w}) = \partial u(\mathbf{w}) = f(s\;(\mathbf{w}))$. This together with the definition $u(\mathbf{w}') = f(s\;(\mathbf{w}')) \cdot \mathbf{w}'$ allows us to conclude that $u(\mathbf{w}) \approx u(\mathbf{w}') + f(s\;(\mathbf{w}')) \cdot \mathbf{w} - f(s\;(\mathbf{w}')) \cdot \mathbf{w}' = f(s\;(\mathbf{w}')) \cdot \mathbf{w}$. The error of this first order approximation is therefore $f(s\;(\mathbf{w}')) \cdot \mathbf{w} - u(\mathbf{w})$ which is exactly the regret $r(\mathbf{w}'|\mathbf{w})$.

This is illustrated in Figure 2 (a). Further, given any two weights $\mathbf{w}', \mathbf{w}'' \in W$, the maximum regret on the line segment L between $\mathbf{w}', \mathbf{w}''$ given $\Omega = \{\mathbf{w}', \mathbf{w}''\}$ occurs at the weight on L coinciding with the intersection of the tangent lines to $u(\mathbf{w})$ at $\mathbf{w}'$ and $\mathbf{w}''$ along L (Figure 2 (b)). In light of this analysis, the objective in (4) is solved by a set Ω that provides the best linear interpolation of the concave function $u(\mathbf{w})$. These insights are leveraged in the following section.
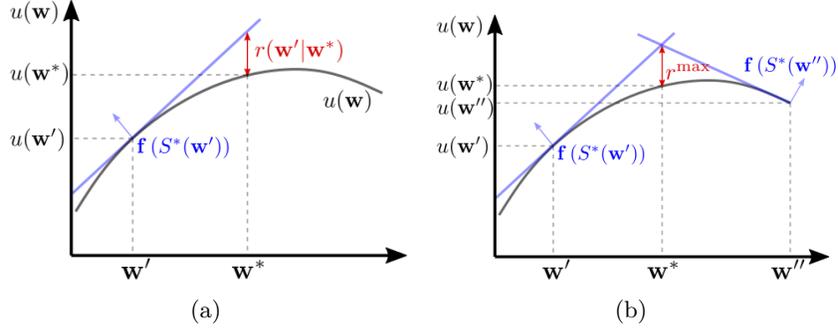
Fig. 2: (a) Regret as the error of a first order approximation. (b) Maximum regret given a set $\Omega = \{\boldsymbol{w}', \boldsymbol{w}''\}$ at the intersection of their tangent lines.

(a) $P(\boldsymbol{w}), \mathcal{C}(N)$ for $N = \{\boldsymbol{w}^1, \boldsymbol{w}^2, \boldsymbol{w}^3\}$ (b) Regret r, and regret bound R
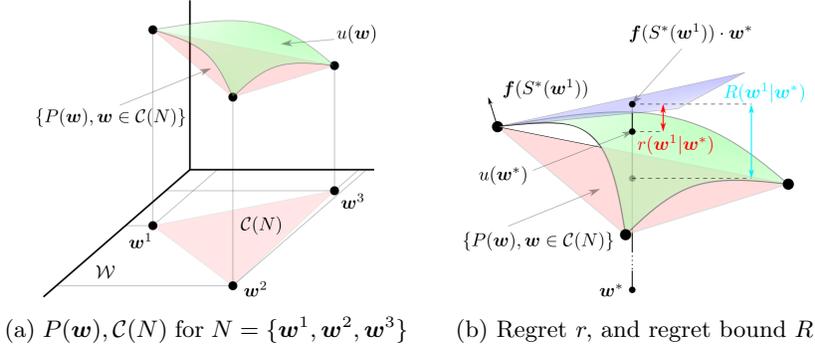
Fig. 3: Illustrative example of a neighborhood and its properties.

4   Algorithm

In this section, we present our solution to Problem 1. The algorithm we propose works by recursively adding weights to a solution set Ω. A strong candidate weight to add is one that is least represented by the current iteration of Ω. The basic framework for such an approach could be described recursively:

$$\Omega^{k+1} = \Omega^k \cup \left\{ \arg\max_{\boldsymbol{w} \in W} \min_{\boldsymbol{w}' \in \Omega^k} r(\boldsymbol{w}'|\boldsymbol{w}) \right\}, \tag{6}$$

where Ω^k is the solution after k iterations from an initial set. Here, (6) recursively adds the weight with the maximum regret given Ω^k. Obtaining the maximizer $\boldsymbol{w}$ is non-trivial due to its nested structure. Instead, our approach replaces $r(\boldsymbol{w}'|\boldsymbol{w})$ in (6) with an upper bound $R(\boldsymbol{w}'|\boldsymbol{w})$ whose maximizer $\boldsymbol{w}$ is obtained from a linear program (LP). Given a set of weights $\Omega \subseteq W$, we define N as a set of n (recall that n is the number of objectives) linearly independent weights $\boldsymbol{w}^1, \ldots, \boldsymbol{w}^n \in \Omega$, and we let $\mathcal{C}(N) \subseteq \mathbb{R}^n$ denote the convex hull of N. We will loosely refer to N as a neighborhood. We define a linear lower bound of the objective value $u(\boldsymbol{w})$ from (1) inside a neighbourhood N. Let $P : W \to \mathbb{R}_{\geq 0}$ be the linear function taking values $P(\boldsymbol{w}^i) = u(\boldsymbol{w}^i)$ for all $\boldsymbol{w}^i \in N$ (Figure 3 (a)).

Error-Bounded Approximation of Pareto Fronts in Robot Planning Problems   9

We denote the difference between the tangent plane at w^ℓ and the P evaluated at w with $R(w^\ell | w) = \mathbf{f}(s(w^\ell))^\top w - P(w)$, $\forall w^\ell \in N, w \in C(N)$. Further, let

$$\bar{R}(N) = \max_{w \in C(N)} \min_{w^\ell \in N} R(w^\ell | w), \quad \bar{w}(N) = \arg\max_{w \in C(N)} \min_{w^\ell \in N} R(w^\ell | w). \quad (7)$$

Finally, we let $\mathcal{F}(N)$ represent the set of feature vectors of the neighborhood:

$$\mathcal{F}(N) = \{\mathbf{f}(s(w^i)), w^i \in N\}. \quad (8)$$

Thus, $R(w^\ell | w)$ is similar to $r(w^\ell | w)$ from (3), but with $u(w)$ replaced with $P(w)$. These definitions, illustrated in Figure 3, motivate the Theorem:

Theorem 2 (Upper Bound of Maximum Regret in a Neighborhood).
Given a neighborhood N of weights, it holds that

$$\max_{w \in C(N)} \min_{w^\ell \in N} r(w^\ell | w) \leq \bar{R}(N).$$

Proof. For any fixed $w \in C(N)$, let $w_1^\ell = \arg\min_{w^\ell \in N} r(w^\ell | w)$ and let $w_2^\ell = \arg\min_{w^\ell \in N} R(w^\ell | w)$. Note that $w_1^\ell = w_2^\ell$. Indeed, we have $R(w_2^\ell | w) \leq R(\hat{w} | w)$ for all $\hat{w} \in N$ if and only if $w^\top \mathbf{f}(s(w_2^\ell)) \leq w^\top \mathbf{f}(s(\hat{w}))$ which is equivalent to $r(w_2^\ell | w) \leq r(\hat{w} | w)$.

By Observation 2, $u(w)$ is concave on $C(N)$ implying that for any $w \in C(N)$, $u(w) \leq P(w)$ (see Figure 3). Thus, by (3), $r(w^\ell | w) \leq R(w^\ell | w)$ for any $w \in C(N)$. The result follows.

$\square$

The value of $\bar{R}(N)$ with corresponding weight $\bar{w}(N)$ from (7) can be obtained by solving the following LP:

$$\max_{x \in \mathbb{R}, w \in \mathbb{R}^n} x - P(w)$$

$$s.t. \begin{bmatrix} f_1(s(w^1)) & \dots & f_n(s(w^1)) & 1 \\ \vdots & \ddots & \vdots & \vdots \\ f_1(s(w^n)) & \dots & f_n(s(w^n)) & 1 \end{bmatrix} \begin{bmatrix} w \\ x \end{bmatrix} \leq \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}, \quad (9)$$

$$w \in C(N).$$

If (x^*, w^*) solves (9), the optimal cost is given by $x^* - P(w^*) = \bar{R}(N)$, and $w^* = \bar{w}(N)$. Indeed, for any feasible x, w, it holds that $x \leq \min_{w^i \in N} \mathbf{f}(s(w^i))^\top w$. Since x is maximized, this will hold with equality for x^*, w^*. Therefore, the cost of (9) is equivalent to $\max_{w \in C(N)} \min_{w^i \in N} R(w^i | w) = \bar{R}(N)$. A detailed explanation of the implementation for Equation (9) is provided in the supplementary materials.

In (9), if $P(w)$ is replaced with $u(w)$, then the resulting problem is solved by x^*, w^* if and only if w^* maximizes the regret in $C(N)$ given the neighborhood N. This problem is not linear and would require solving the LSMOP in (1) potentially many times. Using the LP in (9) our method is summarized in Algorithm 1 described in the next section. We iteratively partition W into smaller neighborhoods, adding weights that result in the largest upper bound of regret.

Algorithm 1: Min-Regret Pareto Sampling (MRPS)

Input: An exact solver to find $s\ (\boldsymbol{w}), \boldsymbol{f}(s\ (\boldsymbol{w}))$; a budget K
Output: Sampled weights Ω and maximum regret

1 $\Omega \leftarrow \{\boldsymbol{e}^i, i = 1, \ldots, n\}$ // where e^i is the ith row of the $n \times n$ identity matrix

2 Obtain $s\ (\boldsymbol{e}^i), \boldsymbol{f}(s\ (\boldsymbol{e}^i)), i = 1, \ldots, n$ from exact solver

3 $\mathcal{N} \leftarrow \{\Omega\}$

4 for $k = n$ *to* K **do**

5 $\quad N = $ neighborhood in $\mathcal{N}$ with maximum $\bar{R}(N)$

6 $\quad$ **if** $\bar{R}(N) = 0$ **then**

7 $\quad\quad$ **break** // Terminate if maximum upper regret bound is 0

8 $\quad \Omega \leftarrow \Omega \cup \{\bar{\boldsymbol{w}}(N)\}$

9 $\quad$ Obtain $s\ (\bar{\boldsymbol{w}}(N)), \boldsymbol{f}(s\ (\bar{\boldsymbol{w}}(N)))$ from exact solver

10 $\quad \mathcal{N} = \mathcal{N} \setminus N$ // Remove max-regret neighborhood

11 $\quad$ **for** $\boldsymbol{w}^i$ *in* N **do**

12 $\quad\quad N^i \leftarrow N \setminus \{\boldsymbol{w}^i\} \cup \{\bar{\boldsymbol{w}}(N)\}$ // Replace w^i with weight of the maximum regret bound

13 $\quad\quad$ **if** N^i *is a neighborhood, i.e., its weights are lin. independent* **then**

14 $\quad\quad\quad \mathcal{N} = \mathcal{N} \cup N^i$

15 $\quad\quad\quad \mathcal{F}(N^i) \leftarrow \mathcal{F}(N) \setminus \{\boldsymbol{f}(s\ (\boldsymbol{w}^i))\} \cup \{\boldsymbol{f}(s\ (\bar{\boldsymbol{w}}(N)))\}$

16 return Ω *and the maximum value of* $\bar{R}(N)$ *over all* $N \in \mathcal{N}$

4.1   Algorithm Description

Algorithm 1 creates and maintains a set $\mathcal{N}$ of neighborhoods $N \subseteq \mathcal{W}$. Each $N \in \mathcal{N}$ is a set of weights $N = \{\boldsymbol{w}^1, \ldots, \boldsymbol{w}^n\}$ where $\boldsymbol{w}^i \in \Omega, i = 1, \ldots, n$. We compute $\bar{R}(N)$ and $\bar{\boldsymbol{w}}(N)$ with the LP in (9) for N using the set of objective vectors $\mathcal{F}(N)$. The algorithm begins with a single neighborhood whose weights are the n canonical basis elements of $\mathbb{R}^n$ (Line 1). The algorithm then iteratively selects the neighborhood N in $\mathcal{N}$ with the largest upper bound of regret (Line 5), and adds its regret weight $\bar{\boldsymbol{w}}(N)$ to Ω (Line 8). It then splits and replaces N with at most n smaller neighborhoods (Lines 11-15) formed by iteratively replacing elements in N with $\bar{\boldsymbol{w}}(N)$ (Line 12). Finally, the algorithm returns the set Ω as well as an upper bound on the regret given Ω (Line 16). Two steps of the algorithm are illustrated in Figure 4, starting with a single neighborhood N_1 in (a) which is then split around $\boldsymbol{w}^3 = \bar{\boldsymbol{w}}(N_1)$ into two new neighborhoods $\mathcal{N} = \{N_2, N_3\}$ in (b). Since $\bar{R}(N_3) > \bar{R}(N_2)$, N_3 is split around $\boldsymbol{w}^5 = \bar{\boldsymbol{w}}(N_3)$ in (c). Finally, in (d), the red area shows the regret given Ω.

Observe that Algorithm 1 may be modified to compute a set Ω given a desired maximum regret $r_{\mathsf{max}} > 0$. This could be accomplished by replacing the input K with r_{max}, and the stopping criteria in Line 4 with a while loop that runs until $\bar{R} \leq r_{\mathsf{max}}$. Here, $\bar{R}$ represents the maximum regret over all neighborhoods $\bar{R} = \max_{N \in \mathcal{N}} \bar{R}(N)$ and can be maintained in the body of the loop. Since $\mathcal{N}$ forms a partition of $\mathcal{W}$, we are guaranteed that the regret of any weight given
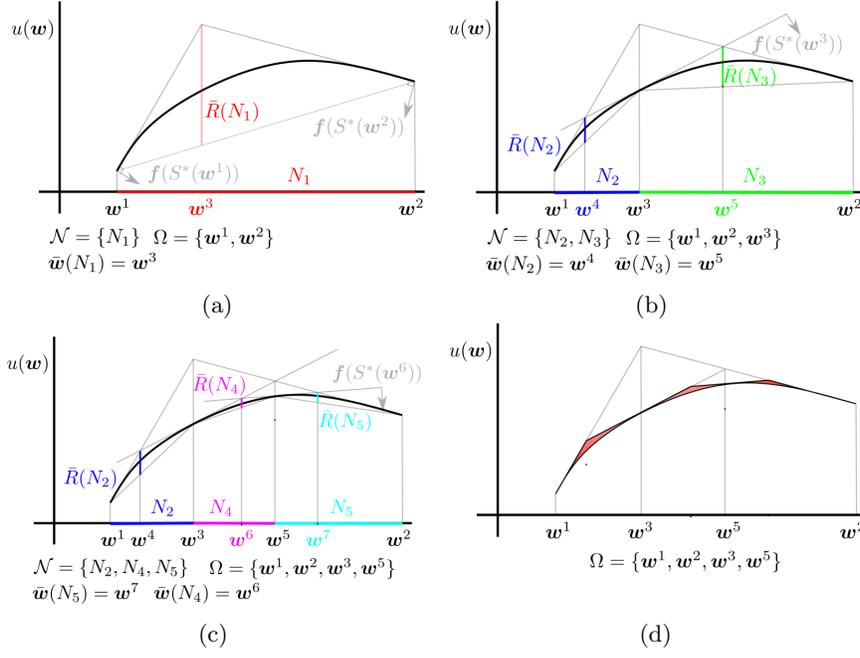
Fig. 4: Illustration of the first two iterations of Algorithm 1.

Ω is no more than $\bar{R}$ by Theorem 2. Therefore, if Algorithm 1 terminates when $\bar{R} \leq r_{\mathsf{max}}$, the desired maximum regret is achieved.

4.2   Algorithm Properties

We observe several beneficial properties to the approach outlined above. First, for a budget of K, Algorithm 1 will require that the LSMOP in (1) be solved at most K times, once per element of Ω. Second, the value of $\bar{R}(N)$ returned by the algorithm is an upper-bound on the value of regret in the original problem (4). Indeed, initially $\mathcal{N} = \{\Omega\}$ and $C(\Omega) = W$. At every iteration, a neighborhood $N \in \mathcal{N}$ is split into at most n sub-neighborhoods such whose convex hulls are disjoint and collectively form $C(N)$. Then, by Theorem 2, it holds that $\max_{\boldsymbol{w} \in W} \min_{\boldsymbol{w}' \in \Omega} r(\boldsymbol{w}' | \boldsymbol{w}) \leq \max_{N \in \mathcal{N}} \bar{R}(N)$.

Further, the set $\Omega(K)$ returned by Algorithm 1 on input K asymptotically and monotonically approaches a set with zero regret as $K \to \infty$. The proof of this is omitted for brevity, but holds because otherwise a neighborhood $N \in \mathcal{N}$ would fail to decrease in size when split (Lines 11-15). This in turns requires that there is a $\boldsymbol{w}^i \in N$ such that $||\bar{\boldsymbol{w}}(N) - \boldsymbol{w}^i||_2$ decreases to 0. Since $\bar{\boldsymbol{w}}(N)$ is chosen to maximize $\bar{R}(N)$, this can only occur if $\bar{R}(N)$ is unbounded at $\boldsymbol{w}^i$ implying that the objectives are unbounded at $\boldsymbol{w}^i$ in violation of Assumption 2.

(a) Approximation using `Uniform`.   (b) Approximation using `MRPS`.
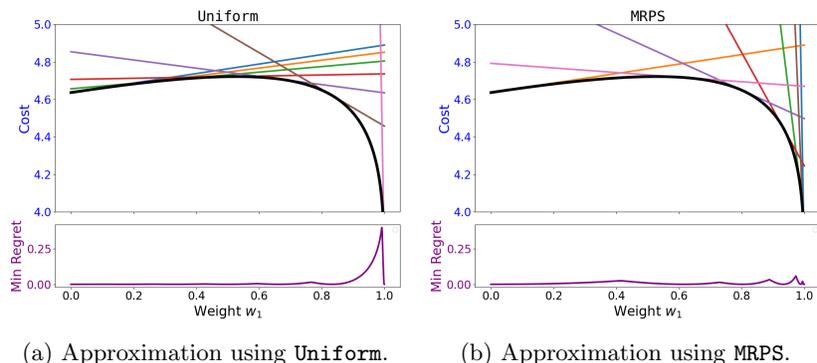
Fig. 5: Approximation of $u(\boldsymbol{w})$ and resulting regret with `Uniform` (a) and `MRPS` (b).

5   Simulation Results

We demonstrate our algorithm in simulations for two different domains: Trajectory planning and learning human preferences. We compare our proposed algorithm to uniform sampling in the weight space [11, 28, 29]. This baseline is denoted by `Uniform`, while the proposed approach from Algorithm 1 is `MRPS`.

5.1   Experiment 1: Dubins Trajectories

In the first experiment, we approximate the Pareto front for a LSMOP motion planning problem for a Dubins vehicle, as shown in the example from Figure 1. The objectives in the problem are trajectory length and integral of the squared (IS) jerk. To find solutions $s\ (\boldsymbol{w})$, the motion planner can compute different Dubins trajectories using different turn radia and pick the optimum among these.

Illustrative Example First, we present more insight into the example in Figure 1 with $K = 7$ samples. We notice that `MRPS` produces a larger variety of sample trajectories, especially those with shorter length. This is also visualized in the approximations of the Pareto front: `Uniform` exhibits a large gap, while the proposed method places samples more homogeneously. In Figure 5 we show the optimal cost $u(\boldsymbol{w})$ (ground truth computed with 10,000 uniform weights), together with the tangent planes of the approximating samples of both approaches. Here, `MRPS` places more samples at the right end where $u(\boldsymbol{w})$ changes more rapidly. Thus, the gap between the best approximating weight and the cost, i.e., the regret, is substantially smaller than `Uniform`.

Quantitative Analysis We repeat the above Dubins planning experiment, but with randomized goal locations and various sampling budgets K. We evaluate algorithm performance on three measures: the regret as defined in (3), the relative regret where the difference in (3) is replaced with a ratio, and the hypervolume of the estimated Pareto front, similar to [23, 24], using an exact integration for 2 features, and a Monte Carlo approximation as in [24] for higher dimensions.
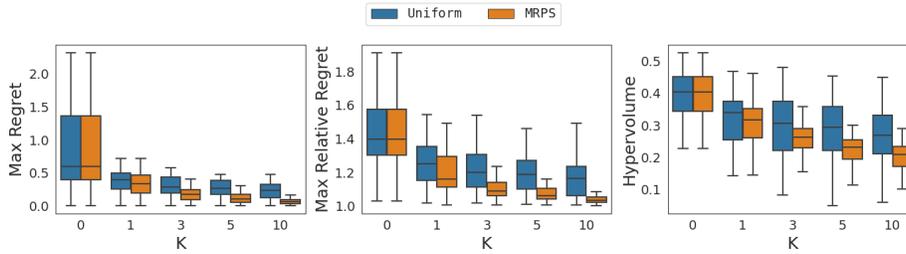
Fig. 6: Numerical Results for the Dubins experiment with 3 objectives.

Moreover, we consider cases with varying numbers of objectives. Figure 6 shows the result for a three objective system consisting of trajectory length, IS jerk and maximum jerk. When $K = 0$ both approaches only use the basis solutions $e^1, \ldots, e^n$, i.e., only the single objective solutions are available. For both, absolute and relative regret, we observe that MRPS achieves substantially smaller values for all $k > 0$. With just 3 samples, MRPS achieves a median regret of .17, corresponding to a median of 1.08 in relative regret, i.e., percent error. In contrast Uniform only achieves a medians of .23 and 1.16 with 10 samples for regret and relative regret, respectively. The smaller estimated hypervolume also suggests that the MRPS-samples lead to a tighter linear approximation of the Pareto front. In summary, the samples found by MRPS for small K have smaller regret and hypervolume than samples found by Uniform with $K = 10$. Thus, our proposed method uses samples much more efficiently.

Varying number of objectives We also considered problem variances with 2 or 4 objectives. As expected, with two objectives (length and IS jerk) both approaches achieve a smaller regret with fewer samples. Nonetheless, the MRPS samples for $K = 3$ are superior to the Uniform samples for $K = 10$, and the regret of MRPS actually converges to 0 for $K = 10$. We also tested the algorithm with four features, where the fourth objective is to avoid part of the environment. While the regret is larger for both approaches, we still observe the same trend that MRPS with 3 samples is on-par with Uniform using 10 samples.

5.2 Experiment 2: Reward Learning

As a second example, we show how using the proposed sample method when learning user preferences, i.e., learning a user specific, but hidden weight vector $\boldsymbol{w}$. As mode of user interaction, we consider learning from choice [4, 8, 11, 12, 29] where the user is iteratively queried with two potential robot trajectories and indicates the preferred one. Repeating this over multiple iterations allows the robot infer about the user weights $\boldsymbol{w}$. Most algorithms for this problem require a set of presampled trajectories from which the best query is selected using some heuristic [8, 11, 12, 29]. These presamples are either random trajectories [8, 12], or optimal solutions for uniformly random weights for the LSMOP [11, 20, 29].

Our proposed algorithm MRPS can be used to generate presamples for these learning problems. Thus, we compare the learning progress over 10 iterations
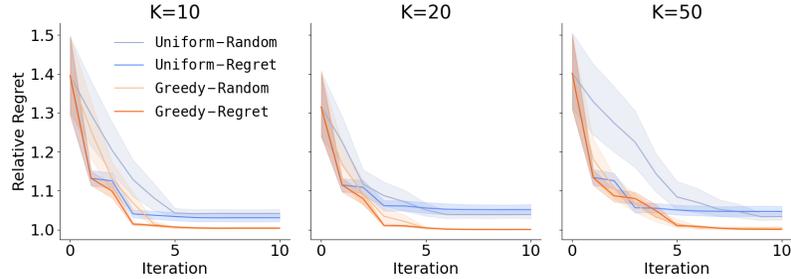
Fig. 7: Learning from choice with presampled sets of different size K. Blue shows learning progress when using uniformly randomly sampled trajectories for active learning with either random or regret queries. Orange shows learning progress for the same query methods, but trajectories are sampled with our proposed algorithm.

when either using `Uniform` or `MRPS` samples. For a clear comparison, we use a simple, deterministic user model: presented with trajectories A and B, they choose A if and only if $f(A)\boldsymbol{w}^* \geq f(B)\boldsymbol{w}^*$. Similar to [4, 26] the trajectory preferred in the previous iteration constitutes one of the two trajectories presented in the next. The robot employs *active* learning: it can choose the second trajectory to be presented in the next iteration. We employ a random selection from the presampled set (`Random`), or the minmax regret approach from [11] (`Regret`). Finally, generating random user weights $\boldsymbol{w}^*$ is not trivial: When $\boldsymbol{w}^*$ is drawn uniformly random, the sample set `Uniform` comes from the same distribution, biasing the experiment. Thus, we randomly select users from the union of two sample sets $\Omega(\text{Uniform})$ and $\Omega(\text{MRPS})$ each of size $K = 20$. We evaluate learning performance using relative regret.

We test this learning framework using the four feature Dubins planning problem from earlier, with one fixed goal location. Figure 7 shows the result for learning with presampled sets of various sizes K. We compare four different approaches: `Uniform` presamples combined with `Random` or `Regret` active learning, and `MRPS` presamples with `Random` or `Regret` active learning. Overall we observe that the `MRPS` samples lead to a smaller learning error than `Uniform` samples, regardless of the query method (`Random` or `Regret`). Indeed, the `Uniform` sets do not always include a close-to-optimal sample such that the learning is eventually unable to make progress. That is the case when $\boldsymbol{w}^*$ was drawn from the `MRPS` samples. However, the opposite effect is negligible: Learning with the `MRPS` samples finds close-to-optimal solutions, implying that the error is very small even when $\boldsymbol{w}^*$ comes from `Uniform`.

When comparing different sizes K of the sample sets, we observe that all approaches learn slightly slower for larger K - the increased number of available trajectories seems to rather distract the learning algorithm than offering more informative queries. More surprisingly, when using `Uniform` samples, the learning still stops making progress. This indicates that the larger set still does not contain close-to-optimal solutions. This further supports our earlier findings that while `MRPS` only needs small K to find a close-to-optimal sample for any $\boldsymbol{w}^*$, while `Uniform` is unable to achieve the same even for large K.

In summary, the experiment shows that using MRPS to generate pre-sampled solutions in reward learning allows for learning close-to-optimal solutions with significantly fewer samples than when relying on randomly generated pre-samples.

6   Discussion and Future Work

In this work, we present a method by which a set of weights can be computed that bounds the maximum regret for any weight in the weight space. However, we only presented results for the case when an exact solver of the underlying LSMOP is available. In future, we will extend our work to the case when an approximation algorithm is used. Further, we constrained our analysis to linear scalarization. If the Pareto front of the underlying MOP is non-convex, linear scalarization fails to capture all Pareto-optimal solutions. However, many of the theoretical results presented here may be extended to Chebyshev scalarization which is known to be Pareto-complete.

References

1. J. Ziegler, P. Bender, T. Dang, and C. Stiller, "Trajectory planning for bertha—a local, continuous method," in *2014 IEEE intelligent vehicles symposium proceedings*, 2014, pp. 450–457.
2. A. Botros and S. L. Smith, "Tunable trajectory planner using G3 curves," *IEEE Transactions on Intelligent Vehicles*, 2022.
3. A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan, "Learning robot objectives from physical human interaction," in *Conference on Robot Learning (CoRL)*. PMLR, 2017, pp. 217–226.
4. N. Wilde, A. Blidaru, S. L. Smith, and D. Kulić, "Improving user specifications for robot behavior through active preference learning: Framework and evaluation," *International Journal of Robotics Research*, vol. 39, no. 6, pp. 651–667, 2020.
5. M. Cáp and J. Alonso-Mora, "Multi-objective analysis of ridesharing in automated mobility-on-demand," in *Robotics: Science and Systems (RSS)*, 2018.
6. R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Structural and multidisciplinary optimization*, vol. 41, no. 6, pp. 853–862, 2010.
7. C.-L. Hwang and A. S. M. Masud, *Multiple objective decision making| methods and applications: a state-of-the-art survey.* Springer Science & Business Media, 2012, vol. 164.
8. D. Sadigh, A. D. Dragan, S. S. Sastry, and S. A. Seshia, "Active preference-based learning of reward functions," in *Robotics: Science and Systems (RSS)*, Jul. 2017.
9. J. Y. Zhang and A. D. Dragan, "Learning from extrapolated corrections," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7034–7040.
10. R. Holladay, S. Javdani, A. Dragan, and S. Srinivasa, "Active comparison based learning incorporating user uncertainty and noise," in *RSS Workshop on Model Learning for Human-Robot Communication*, 2016.
11. N. Wilde, D. Kulić, and S. L. Smith, "Active preference learning using maximum regret," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 10 952–10 959.

12. E. Biyik, M. Palan, N. C. Landolfi, D. P. Losey, and D. Sadigh, "Asking easy questions: A user-friendly approach to active reward learning," in *Conference on Robot Learning (CoRL)*, 2019.

13. J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *IEEE Intelligent Vehicles Symposium*, 2011, pp. 163–168.

14. O. Schütze, O. Cuate, A. Martín, S. Peitz, and M. Dellnitz, "Pareto explorer: a global/local exploration tool for many-objective optimization problems," *Engineering Optimization*, vol. 52, no. 5, pp. 832–855, 2020.

15. D. Dolgov, S. Thrun, M. Montemerlo, and J. Diebel, "Path planning for autonomous vehicles in unknown semi-structured environments," *International Journal of Robotics Research (IJRR)*, vol. 29, no. 5, pp. 485–501, 2010.

16. Y. Zeng, X. Xu, S. Jin, and R. Zhang, "Simultaneous navigation and radio mapping for cellular-connected uav with deep reinforcement learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 7, pp. 4205–4220, 2021.

17. D. Kent and S. Chernova, "Human-centric active perception for autonomous observation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 1785–1791.

18. M. Cakmak, S. S. Srinivasa, M. K. Lee, J. Forlizzi, and S. Kiesler, "Human preferences for robot-human hand-over configurations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011, pp. 1986–1993.

19. C. Basu, M. Singhal, and A. D. Dragan, "Learning from richer human guidance: Augmenting comparison-based learning with feature queries," in *Proceedings of the ACM/IEEE HRI*, 2018, pp. 132–140.

20. N. Wilde, E. Biyik, D. Sadigh, and S. L. Smith, "Learning reward functions from scale feedback," in *Conference on Robot Learning (CoRL)*. PMLR, 2022, pp. 353–362.

21. A. Bobu, D. R. Scobee, J. F. Fisac, S. S. Sastry, and A. D. Dragan, "Less is more: Rethinking probabilistic models of human behavior," in *Proceedings of the ACM/IEEE HRI*, 2020, pp. 429–437.

22. V. Pereyra, M. Saunders, and J. Castillo, "Equispaced pareto front construction for constrained bi-objective optimization," *Mathematical and Computer Modelling*, vol. 57, no. 9-10, pp. 2122–2131, 2013.

23. J. Xu, A. Spielberg, A. Zhao, D. Rus, and W. Matusik, "Multi-objective graph heuristic search for terrestrial robot design," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 9863–9869.

24. S. Parisi, M. Pirotta, and J. Peters, "Manifold-based multi-objective policy search with sample reuse," *Neurocomputing*, vol. 263, pp. 3–14, 2017.

25. J. Lee, D. Yi, and S. S. Srinivasa, "Sampling of pareto-optimal trajectories using progressive objective evaluation in multi-objective motion planning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

26. N. Wilde, A. Sadeghi, and S. L. Smith, "Learning submodular objectives for team environmental monitoring," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 2, pp. 960–967, 2021.

27. S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

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

29. N. Wilde, D. Kulić, and S. L. Smith, "Bayesian active learning for collaborative task specification using equivalence regions," *IEEE Robotics and Automation Letters (RA-L)*, vol. 4, no. 2, pp. 1691–1698, Apr. 2019.