MO-BBO: Multi-Objective Bilevel Bayesian Optimization for Robot and Behavior Co-Design

Yeonju Kim[1], Zherong Pan[2] and Kris Hauser[2]

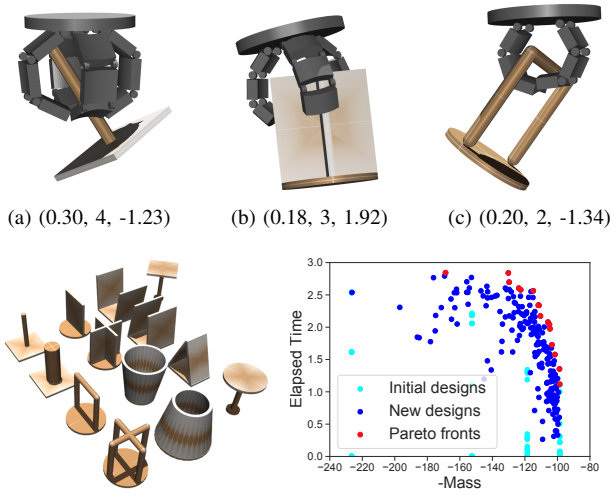
***Abstract*— Robot design is a time-consuming process involving repeated experiments in a variety of environments to optimize multiple, possibly conflicting performance metrics. Moreover, the optimal robot performance for a given design depends on how the robot adapts its behavior to its environment. We propose a multi-objective Bilevel Bayesian optimization (MO-BBO) technique to automate the process of form-behavior co-design. The approach expands the Pareto front of multiple metrics by simultaneously exploring the robot design and behavior. MO-BBO uses a bilevel optimization of the acquisition function with design and behavior parameters being the high- and low-level decision variables, respectively. In the low-level, we always choose environment-aware behaviors that maximize each metric. We evaluate MO-BBO in applications to grasping gripper design and bimanual arm placement, and show that our method can efficiently focus samples on the Pareto front and generate a diversity of designs.**

I. Introduction

Robot design is difficult even for skilled engineers because a robot's performance is determined both by design and runtime behavior in different environments. Hence, to completely understand the impact of a given design choice, the designer must consider how the robot may adapt its behavior to different environments, either exploiting favorable impacts of the design choice or mitigating negative impacts. This process requires tremendous trial and error and close collaboration with behavior software developers. Design automation tools are potentially useful for assisting designers, but classical tools cannot be applied directly because they either ignore one of the design and behavior parameters, or do not model environment-dependence, leading to active research in simultaneous robot and behavior co-design optimization.

In traditional robot design process, the robot's design and behavioral parameters are determined in two stages. During the design time, we define the robot's design parameters, e.g. kinematics, materials, and actuator parameters. During the test time, the robot's performance is evaluated in a number of environments, and for each environment and design, the behavior is chosen separately. Further, multiple conflicting performance metrics are used to evaluate the robot's performance, and the designer does not possess a clear preference over metrics.

Prior robot optimization algorithms have failed to model the two-stage, three-way coupling between the robot's design, behavioral parameters, and the environments. For example, the evolutionary optimization [6] jointly searches for design and behavioral parameters but only for a single task in a single environment. This method also requires a vast number of experiments, which is not tractable if experiments are costly. Similarly, topological robot shape and controller can be optimized simultaneously guided by a local gradient in [1], but they are optimized in a single stage instead of two stages. In addition, gradient-based methods are incapable of handling uncertainty and non-smooth objective functions. Contextual Bayesian optimization (CBO) [4] or contextual bandit algorithm [16] are environment-aware but do not address the multi-objective co-design problem. Finally, Bayesian Optimization (BO) has been modified to optimize both design and behavior parameters [15], but their method does not consider multi-environments or multi-objective settings.



(a) (0.30, 4, -1.23) (b) (0.18, 3, 1.92) (c) (0.20, 2, -1.34)

(d) Object set (e) Metric space explored by MO-BBO

Fig. 1: Gripper design problem, with (a–c) showing example designs with parameters (finger length (m), #fingers, finger curvature (m^{-1})). (d): The set of graspable objects. (e) Metric values of 200 gripper designs explored by MO-BBO, of which 16 designs are on the Pareto front.

We propose a Multi-Objective Bilevel Bayesian optimization (MO-BBO) algorithm that accounts for all the features of the robot / behavior co-design problem. BO minimizes the number of costly metric evaluations by approximating the mapping from design & behavior variables to performance metrics via a cheaper Gaussian process surrogate model. At each iteration, it proposes new design candidates by maximizing an acquisition function that depends on the surrogate model. This acquisition function is selected to grow the Pareto front of the achievable performance metrics. Our main contribution is a bilevel optimization of the acquisition

*Y. Kim and K. Hauser are partially supported by NSF Grant #IIS-2002492. [1]Y. Kim is with the is Dept. of Mechanical Science and Engineering, University of Illinois at Urbana-Champaign, Urbana, IL, USA yeonjuk2@illinois.edu. [2]Z. Pan and K. Hauser are with the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL, USA {zherong,kkhauser}@illinois.edu.

function, where design parameters are optimized at the high-level, which is followed by optimizing the behavior parameters on the low-level for each environment.

We employ this framework in two problem domains: grasping gripper design and arm placement for a bimanual mobile manipulator. Experiments demonstrate that the framework is able to efficiently disregard ineffective designs and focus its efforts on expanding the Pareto front. Moreover, it is able to generate a diversity of designs in which each design obtains high scores along one or more metric dimensions.

II. Related Work

Robot design optimization plays a role in many applications including optimization of gaits and policies [15], [25], geometric shape [1], [17], reconfigurable and modular robots [12], [28], and actuation mechanisms [5].

Local algorithms update the robot design and behavior in a neighborhood of an initial guess. Guided by gradient information, these methods are relatively fast and scalable to high dimensions. However, these typically use application-dependent algorithms that do not generalize. Topology optimization [1], [27] based on the Method of Moving Asymptotes (MMA) and its variants [23] has found applications in robot link-shape design. The policy gradient method is a form of local optimization usually applied to behavior parameters, but it can be augmented to optimize both design parameters and behavior [11]. However, these approaches yield only local improvements and cannot easily generalize to optimize behavior over many environments.

To address global optimization, search-based methods may be applied to discrete design spaces but with continuous behaviors [12], [28]. In mutually continuous design and and behavior space where experimental costs are low, evolutionary algorithms [7] have been used for designing soft robots [6], legged walkers [19], and grippers [20].

Bayesian optimization (BO) approaches have applied to co-design problems with great success. Liao et al apply a hierarchical Bayesian design and behavior factorization to design a legged robot structure and its behavior policies, using sparse evaluations from real experiments [15]. This is the same general bilevel framework as in our method, but we address the cases of multi-objective optimization with environment-dependent behaviors. Conditioning the behavior parameters on the environment or task in BO is known as contextual BO, and has been considered in contextual policy search for (single-objective) robot reinforcement learning tasks [18], but has not, to our knowledge, been used in multiobjective co-design problems.

Our approach guides the co-design optimization problem using techniques from the multi-objective BO literature. Multi-objective BO has been used in robotics for gait policy optimization [2], [24] but prior work is non-contextual and has not been applied to co-design problems. To our knowledge, ours is the first work for mechanism / behavior co-design that is simultaneously multi-objective and contextual.

III. Problem Definition

We assume that a designer establishes a design space D, a finite set of environments E, and a behavioral space Θ. Behaviors are specified either using a parameterized control policy or trajectory. The designer also establishes one or more performance measures, some or all of which depend jointly on the design and behavior. The goal of our framework is to generate a limited set of candidate designs that spans the Pareto front in the metric space.

A. Performance measures

A performance measure (hereby referred to as a "measure") is defined as a numerical score of the fitness of a design and/or behavior along some criterion that may depend on the chosen environment. Our convention assumes that larger measure values correspond to better designs. Measures come in two forms: 1) *design measures*, where the score is only dependent on the design, such as the mass of a gripper, and 2) *behavior measures*, where the score is evaluated as a function of the environment ($e \in E$, E finite) as well as the chosen design ($d \in D$, D continuous) and behavior ($\theta \in \Theta$, Θ continuous), such as the grasp quality of the gripper for a given object. We assume N design measures $\{f_n(d) | n = 1, \ldots, N\}$ and M behavior measures $\{g_m(d, e, \theta) | m = 1, \cdots, M\}$.

B. Mapping behavior measures to design space

The primary optimization will take place in a multi-objective design metric space. We map behavior metrics to design space as follows. Given a design and an environment, the robot can choose an optimal behavior to maximize a behavior measure by solving the following optimization:

$$\theta^\star_{d,e,m} \triangleq \operatorname*{argmax}_{\theta \in \Theta} g_m(d, e, \theta). \tag{1}$$

Provided with optimized behaviors for each measure and environment, we can factor out the environment variables. One simple approach is to take the average of $g_m(d, e, \theta^\star_{d,e,m})$ over every $e \in E$. However, we observe that designers can prioritize some environments over others. Designers may also categorize environments into groups and measure the performance of each group as separate metric functions. To address these use cases, we introduce a designer-defined weighting matrix W with P rows and $|E|$ columns. For example, when designing a grasping gripper, there can be five different objects to be grasped ($|E| = 5$), the first 2 being cylinders and the last 3 being boxes of different shapes, then W could be set as:

$$W = \begin{bmatrix} 1/2 & 1/2 & 0 & 0 & 0 \\ 0 & 0 & 1/3 & 1/3 & 1/3 \end{bmatrix}, \tag{2}$$

to measure the two kinds of objects separately. The $M|E|$ behavior measures are mapped to PM design measures

$$f_{N+M(j-1)+m}(d) = W_j \begin{bmatrix} g_m(d, e_1, \theta^\star_m(d, e_1)) \\ \vdots \\ g_m(d, e_{|E|}, \theta^\star_m(d, e_{|E|})) \end{bmatrix}, \tag{3}$$

where W_j is the jth row of W. In summary, given behaviors we can define $R = N + PM$ design measures, with the first N being behavior-independent and the last PM being behavior-

and environment-dependent. This yields the following definition of a metric vector, for which larger values are preferable.

***Definition** 3.1:* The *design metric* $\mathbf{f} : D \to \mathbb{R}^{N+PM}$ maps a design to a *metric vector* $\mathbf{f}(d) = (f_1(d), \ldots, f_R(d))$.

Note that from the perspective of the high-level optimization the metric is only dependent on design d, but the metric vector is also dependent on the decisions chosen for each environment and behavior measure. If an algorithm chooses suboptimal decisions then its calculated metric vector will be an underestimate.

C. Pareto Front

Multi-objective problems have a set of solutions that are plausibly "high quality" in the Pareto sense. For a Pareto optimal design, one metric cannot be increased without decreasing another. Formally, this is expressed as the following Pareto dominance relation:

***Definition** 3.2:* For two designs $a, b \in D$ and a metric $\mathbf{f}$, we say a *dominates* b or $a \succ b$ if and only if $f_i(a) \geq f_i(b) \forall i = 1, \ldots, R$ and $f_i(a) > f_i(b)$ for some i. A design a is *non-dominated* with respect to a set $\mathcal{S} \subseteq D$ if no design $b \in \mathcal{S}$ dominates a. A *Pareto optimal* design $a \in D$ is non-dominated with respect to D. The set of Pareto optimal designs is denoted as $D_{\mathbf{PO}}$. The image of the set $D_{\mathbf{PO}}$ is called the *Pareto front* $\mathbf{PF} = \{\mathbf{f}(d) | d \in D_{\mathbf{PO}}\}$.

IV. MO-BBO

Most metrics have no simple closed-form, but rather must be evaluated through simulation or some costly numerical computation. BO is an efficient way to optimize unknown functions with high evaluation costs. BO learns probabilistic *surrogate functions* to approximate the performance measures and sequentially generates designs based on the surrogates. At the tth iteration, we select the design d_t that maximizes the *acquisition function* $\alpha(d)$, which evaluates the design's utility as estimated by the surrogate models. We then evaluate the metrics for design d_t, add the result to the dataset, re-fit the surrogate functions, and repeat.

Multi-objective BO models a separate surrogate function for each objective, and chooses candidates to evaluate based on the optimization of an acquisition that promotes expansion of the Pareto front towards the currently non-dominated region. One natural acquisition function is the expected hypervolume increase [8], but its analytic evaluation is complex. Recent works have shown that many simpler uncertainty-aware acquisition functions, such as uncertainty volume of GP-(L)UCB [3] and predictive entropy [13], can also encourage exploration of the Pareto front. We use a function called Likelihood of Pareto Expansion (LPE), which is relatively computationally inexpensive yet still yields good performance.

In co-design, behavior variables should be optimized for each metric and environment, so that maximizing the acquisition function $\alpha(d)$ over design candidates becomes a bilevel optimization problem. We accommodate the inner optimization in this setting using surrogate models that capture design, environment, and decision-dependent effects.

A. Surrogate Functions

We use Gaussian Processes (GP) to model the surrogate functions. GP gives us the conditional distribution of the objective on points that have not been evaluated. The conditional distribution of function values is used to select the next design. GP approximates N design measures and M behavior measures for each of E environments. The total number of GPs needed is $(N + M|E|)$:

$$f_n(d) \sim \mathbf{GP}_n(d; \mathcal{B}_n) \tag{4}$$

$$g_m(d, e, \theta) \sim \mathbf{GP}_{m,e}([d, \theta]; \mathcal{B}_{m,e}). \tag{5}$$

The notation $y_*(x) \sim \mathbf{GP}_*(x; \mathcal{B}_*)$ indicates that at the query point x, y_* is assumed to take on a Gaussian distribution given the dataset $\mathcal{B}_*$ of prior input-output pairs $\mathcal{B}_* = \{\langle x^{(i)}, y_*^{(i)} \rangle | i = 1 \ldots, n\}$. Specifically, given a covariance function $k(\cdot, \cdot)$, the GP posterior is given by $y_* \sim \mathcal{N}(\mu_*(x), \sigma_*(x))$ where:

$$\mu_*(x) = \mathbf{k}_{*,x}^T \mathbf{K}_{*,*} \mathbf{y} \tag{6}$$

$$\sigma_*^2(x) = k(x, x) - \mathbf{k}_{*,x}^T \mathbf{K}_{*,*}^{-1} \mathbf{k}_{*,x}, \tag{7}$$

and $\mathbf{y} = [y_*^{(1)}, y_*^{(2)}, \ldots, y_*^{(n)}]^T$ is the set of observations, $\mathbf{K}_{*,*}$ is an $n \times n$ matrix of kernel values with $\mathbf{K}_{i,j} = k(x^{(i)}, x^{(j)})$, and $\mathbf{k}_{*,x} = [k(x^{(1)}, x), \ldots, k(x^{(n)}, x)]^T$ is a vector of kernel values evaluated between $\mathcal{B}_*$ and x. The GP fitting step takes $\mathcal{B}_*$ as input, optimizes the covariance function parameters, and precomputes the matrix inverse in (6) and (7).

B. Acquisition Function & Candidate Optimization

The Likelihood of Pareto Expansion (LPE) metric scores a potential design by estimating the probability that it is non-dominated given the surrogate function estimates. We estimate it in Monte-Carlo fashion as follows.

Consider a candidate design $d \in D$. For each environment-independent metric f_n, $n = 1, \ldots, N$, we directly evaluate the mean $\mu_n(d)$ and variance $\sigma_n^2(d)$ of $\mathbf{GP}_n$. For behavior metric g_m, we select a design $\theta_{d,e,m}^*$ and then evaluate the mean $\mu_{m,e}(d)$ and variance $\sigma_{m,e}^2(d)$ of $\mathbf{GP}_{m,e}$ at $[d, \theta_{d,e,m}^\star]$. For each weight matrix row W_j, the mean and variance of the metric $f_{N+M(j-1)+m}$ are $W_j[\mu_{m,e_1}, \ldots, \mu_{m,e_{|E|}}]^T$ and $W_j[\sigma_{m,e_1}^2, \ldots, \sigma_{m,e_{|E|}}^2]^T$, respectively. Overall, this gives us a mean μ and diagonal covariance matrix Σ for the metric space vector $\mathbf{f} \sim \mathcal{N}(\mu, \Sigma)$. To evaluate the LPE, we draw K samples $\{\mathbf{f}^{(k)}(d) \,|\, k = 1, ..., K\}$ from the posterior Gaussian distribution and $\alpha(d)$ is the ratio of the number of non-dominated samples by the current Pareto front $\mathbf{PF}$:

$$\alpha(d) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}[\mathbf{f}^{(k)}(d) \succ \mathbf{PF}]. \tag{8}$$

Selecting $\theta_{d,e,m}^*$ is pivotal to the performance of BBO. Inspired by [21], we select $\theta_{d,e,m}^*$ that optimizes upper bound of a confidence interval of $\mathbf{GP}_{m,e}$:

$$\theta_{d,e,m}^* = \operatorname*{argmax}_\theta \mu_{m,e}(d, \theta) + \kappa \sigma_{m,e}(d, \theta), \tag{9}$$

where $\kappa \geq 0$ trades off exploration and exploitation. Eqn. 9 can be interpreted as a surrogate optimization of Eqn. 1. Putting things together, BBO finds the next design candidate

Algorithm 1 MO-BBO

Input: Iteration count T, initial data set $\mathcal{B}_n, \mathcal{B}_{m,e}$
Input: Upper-level sample size S, LPE sample size K
Input: Exploration-exploitation trade off κ
1: Fit $\mathbf{GP}_n$ from $\mathcal{B}_n$ and $\mathbf{GP}_{m,e}$ from $\mathcal{B}_{m,e}$
2: Set Pareto front $\mathbf{PF}$ from initial data
3: **for** iteration $t = 1, \ldots, T$ **do**
4: Randomly sample S designs D_t uniformly from D
5: **for** $d \in D_t$, $e \in E$, $m = 1, \ldots, M$ **do**
6: Solve $\theta^*_{d,e,m}$ from Eqn. 9
7: Compute $\mu(d)$, $\Sigma(d)$ from GPs and $\theta^*_{d,e,m}$
8: $d_t \leftarrow \text{argmax}_{d \in D_t} \alpha(d)$ (Eqn. 8)
9: Evaluate measures $f_n(d_t), g_m(d_t, e, \theta^*_{d_t,e,m})$
10: Update Pareto front $\mathbf{PF}$
11: Add $\langle d_t, f_n(d_t) \rangle$ to $\mathcal{B}_n$
12: Add $\langle [d_t, \theta^*_{d_t,e,m}], g_m(d_t, e, \theta^*_{d_t,e,m}) \rangle$ to $\mathcal{B}_{m,e}$
13: Re-fit $\mathbf{GP}_n$ from $\mathcal{B}_n$ and $\mathbf{GP}_{m,e}$ from $\mathcal{B}_{m,e}$
14: **return** $\mathbf{PF}$

by performing the following bilevel optimization:

$$\underset{d, \theta^*_{d,e,m}}{\text{argmax}}\ \alpha(d) \quad \text{s.t. Eqn. 9 holds.} \tag{10}$$

C. Algorithm Pipeline

MO-BBO (Alg. 1) starts by initializing the surrogate functions. We draw a set of random design and behavior samples and evaluate the metric vectors. The surrogate functions and initial Pareto front are then trained with this data before the outer loop (lines 1–2). Each outer iteration attempts to find the globally optimal solution to Eqn. 10. We approximate the solution to the high-level problem by restricting the design space to a sampled dataset of size S. For the low-level problem (lines 5–6) we solve for each $\theta^*_{d,e,m}$ usnig an optimization of Eqn. 9. In our experiments we compare two methods: brute force uniform random sampling, and a gradient-free global optimization method (DIRECT). Lines 7–13 are the same as a standard multiobjective Bayesian Optimization, except that the decision metrics and their surrogate functions are updated separately.

D. Performance Discussion

Let us assume that C is the cost of evaluating each metric, which may be rather high in general. We analyze the computational cost of Alg. 1. At the tth iteration, the cost of evaluating a surrogate function is $O(t)$. If $\theta^*_{d,e,m}$ are known, the cost of evaluating the acquisition function is $O((N+M|E|)Kt)$ because $\mathbf{PF}$ may contain $O(t)$ points and K samples are drawn to evaluate $\alpha(d)$. Re-fitting a surrogate function costs $O(t^3)$. Excluding the low-level optimization, the running time of each iteration is $O((N+M|E|)(KtS + C + t^3))$. Note that the t^3 re-fitting term involves small constant factors, and can also be reduced to t^2 using efficient GPU implementations of GPs ([10]). In Alg. 1, low-level optimization is solved for $SM|E|$, which incur a cost of $O(SM|E|J)$, which tends to dominate complexity. Here J is the cost of solving global optimization using DIRECT. The overall running time of each iteration of Alg. 1 is thus $O((N+M|E|)(KtS+C+t^3)+SM|E|J)$ and the total running time is $O((N+M|E|)T(KTS+C+T^3)+TSM|E|J)$.

E. Subsampling

As a postprocessing step, we select a representative subset of design candidates from the computed $\mathbf{PF}$. This is important because BBO may return a large number of Pareto-optimal designs, many of which exhibit similar performance characteristics. The sub-sampling step provides users with a manageable number of design candidates with a relatively uniform distribution across $\mathbf{PF}$. Our subsampling algorithm uses a heuristic for solving the p-dispersion problem called Greedy construction heuristic [9]. Let the Pareto optimal designs and their metric values be $\langle d^{(1)}, \mathbf{f}^{(1)} \rangle, \ldots, \langle d^{(Q)}, \mathbf{f}^{(Q)} \rangle$. We select p out of Q designs to form our subset. The subset is initialized by choosing two farthest points in $\mathbf{PF}$ as measured by their weighted Euclidean distance between metric vectors, with each dimension normalized to the range [0,1]. During each iteration, we add the point that maximizes the minimum distance to the points already in the subset. This continues until the subset contains p points.

V. Experiments

Using two exemplary problems, we evaluate how well BBO returns a Pareto front $\mathbf{PF}$ that populates the design space. We use two metrics, HyperVolume (HV) and Metric Spread (MS) (larger values imply beter performance). HV measures the volume of points $\mathbf{f} \leq \mathbf{f}^\star$ that are dominated by $\mathbf{PF}$, with respect to a lower origin $\mathbf{f}^\star$. MS measures the gap between the minimum and maximum metric value of points in $\mathbf{PF}$, for each metric dimension $i = 1, \ldots, R$. There are R such gaps, each given by $\max_{d \in D_{\mathbf{PO}}} f_i(d) - \min_{d \in D_{\mathbf{PO}}} f_i(d)$.

A. Gripper Design Problem

We first consider a gripper design problem in which the goal is to reduce gripper mass while improving the robustness of grasping a diverse range of objects. The design parameters include finger shape and count, and our object set consists of 13 objects with varying shapes and difficulty of grasping as shown in Fig. 1 (a). Our object set is created using geometric primitives to represent a wide range of grasp types. This is a relatively small-scale design problem that we use to explore the effect of algorithm parameters on overall performance. Nevertheless, each optimization takes on the order of a few hours to complete.

This problem designates 3 design parameters d: finger length (range [0.1 m,0.4 m]), num fingers (2, 3, or 4), and finger curvature (range [-3,3], in m^{-1}). These are illustrated in Fig. 1 (bcd). The 3 behavior variables indicate the approach orientation of a grasping motion and gripper's rotation along z-axis, designated by the angles $\theta \triangleq (\theta, \phi, \beta)$. The gripper approaches the center of the object from a standoff distance 3m in direction $(-\cos\theta\cos\phi, -\sin\theta\cos\phi, -\sin\phi)$. ϕ has range $[\frac{\pi}{4}, 0.99\frac{\pi}{2}]$ so that the gripper starts above the table.

Metrics include the *reciprocal of the mass* of the gripper $f_1(d) = \text{Gripper-Mass}(d)^{-1}$ (design metric) and elapsed time g_1, which measures the time elapsed until the gripper drops an object during lifting and shaking (behavior metric). To evaluate g_1, we simulate the gripper approaching, grasping, lifting with a 0.5 s vertical motion, and finally shaking objects

Object	1	2	3	4	5	6	7	8	9	10	11	12	13
Design 1 Mass 168.8													
Elapsed-Time	2.274	3.0	2.908	3.0	2.347	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Design 2 Mass 130.2													
Elapsed-Time	2.521	3.0	3.0	3.0	3.0	3.0	2.427	1.641	3.0	3.0	3.0	3.0	3.0
Design 3 Mass 111.7													
Elapsed-Time	0.5	3.0	2.9	3.0	3.0	2.409	0.5	1.564	1.686	1.278	3.0	3.0	3.0
Design 4 Mass 98.53													
Elapsed-Time	3.0	3.0	3.0	2.145	1.360	1.251	0.	0.	0.	0.	1.205	0.	0.

TABLE I: Four gripper designs from the Pareto Front of Fig. 1e. Images captured during lifting and shaking. Elapsed-Time indicates how long the object stays held during shaking, in seconds (max 3 s).

Metrics	Baseline	Sample size (S)			Confidence Bound (κ)				#func. evals. ($\max f$)		
	$(50, 2.0, 1k)$	100	20	10	0.0	1.0	5.0	10.0	10000	100	10
Inner Optimization via Uniform sampling / DIRECT											
HV	2.703/2.746	2.646/2.693	2.684/2.759	2.713/**2.776**	**2.784/2.871**	2.753/**2.787**	2.580/2.706	2.438/2.543	2.694/2.747	2.624/2.749	2.502/2.553
MS(100/Gripper-Mass)	0.328/0.346	0.248/0.299	0.371/0.290	0.346/**0.391**	0.245/0.209	0.242/0.329	0.320/0.331	**0.436/0.395**	0.281/0.321	0.265/0.308	0.296/0.374
MS(Elapsed-Time)	1.739/**1.793**	1.659/1.718	**1.758/1.790**	**1.755/1.822**	1.748/**1.825**	1.729/**1.773**	1.658/**1.797**	1.526/1.656	1.720/**1.764**	1.632/**1.802**	1.543/1.621

TABLE II: The hypervolume (HV) and metric spread (MS) metrics of optimization performance at various parameter settings, averaged over four trials. The reference point used for HV computation is $(0, 0)$. Bold indicates the top 25% of the observed range.

along a sinusoidal wave with amplitude 0.2m and period 0.5 s for 5 periods. We halt the simulation once the object slips off the gripper and record the elapsed time. We set W to $\left[\frac{1}{|E|} \quad \cdots \quad \frac{1}{|E|}\right]$, so that the ultimate metric space is 2D:

$$f_1(d) = \text{Gripper-Mass}^{-1}(d) \text{ and}$$
$$f_2(d) = \frac{1}{|E|} \sum_{e \in E} \text{Elapsed-Time}(d, e, \theta^\star_{d,e}). \tag{11}$$

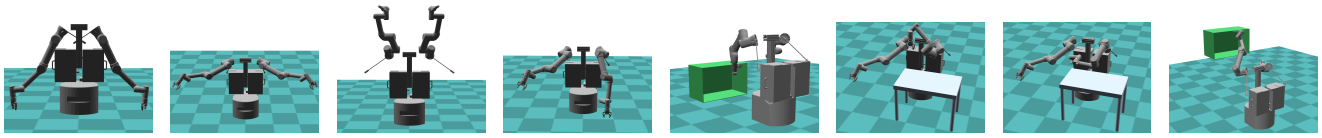
Fig. 1e illustrates the metric space explored by the algorithm under settings ($S = 20, \kappa = 2, \max f = 1,000$). After initialization, few poor designs are generated and most of the optimization effort is focused near the Pareto front. Each optimization takes between 1.5–3 hours. Tab. I examines some of the optimal gripper designs in the Pareto front in more detail. The Elapsed-Time metric for each of the objects is shown with a representative grasping configuration.

To evaluate the effects of parameter settings, we use $(S, \kappa, \max f) = (50, 2.0, 1000)$ as a baseline, and evaluate the difference performance by changing each parameter. Results in Tab. II are averaged over four trials to account for some randomness. Overall, performance is relatively insensitive to parameter settings. The metric HV and MS tend to increase as S decreases. As κ increases, the MS for the mass metric increases, but the HV and MS for the elapsed time metric decrease. Changing $\max f$ does not affect the Pareto front significantly, but a larger value increases computation time. DIRECT generally performs slightly better than uniform sampling for the inner optimization, but DIRECT takes slightly longer computation (2.97 h vs 1.66 h).

B. Robot Arm Design Problem

Next, we consider a real-world problem where a dual-arm mobile manipulator is designed for human environments, such as offices and hospitals. Two UR5 robot arms are symmetrically mounted on a torso, and the design problem is to choose the mounting orientation and shoulder width of the arms to maximize reachability of grasping targets in typical human environments. We designate 3 design parameters: distance between shoulder (range [0.4 m, 0.65 m]), pan angle (range [0°, 90°]), and tilt angle (range [0°, 180°]). We define 5 manipulation scenarios (Fig. 2. d–h) for the left arm only: 1) Ground, 2) Low-shelf, 3) Table-horizontal, 4) Table-vertical, and 5) High-shelf. In each scenario, a discretized uniform grid of position targets are defined. We also choose a few scenario-dependent goal orientations to reach. In Ground and Table-vertical, the gripper must meet a single vertical orientation. For Low-shelf and High-shelf, we consider 3 horizontal orientations with {-60°, 0°, 60°} yaw. For Table-horizontal, we consider 6 horizontal directions with yaws evenly spaced from 0° to 360°.

For performance evaluation, we use one behavior measure g_1, which measures the mean reachability index over all target positions designated by e (similar to the reachability index used in [26], defined as the fraction of reachable orientations). The behavior parameters consist of 6 joint angles of the left arm used as the IK seed for reachability testing, and a local optimization (Newton-Raphson) method is used to reach each 6DOF pose in the bounding box. The number of GPs used is $N + M|E| = 5$. To inspect all behavior metrics, we set $W = I_{5\times5}$, giving a 5D metric space: $f_i(d) = \text{Avg-Reachability}(d, e_i, \theta^\star_{d,e_i,1}), i = 1, \ldots, 5$.



(a) (0.4, 0, 35) (b) (0.5, 30, 80) (c) (0.65, 80, 130) (d) Ground (e) Low-shelf (f) Table-vertical (g) Table-horiz. (h) High-shelf

Fig. 2: Robot arm design problem. (a–c) show 3 illustrative designs with design parameters (shoulder width (m), pan (°), tilt (°)). (d–h) show test scenarios.

Scenario	Ground	Low-shelf	Table-vert	Table-horiz	High-shelf
Legend (# of target orientation reached)	0 1	0 3	0 1	0 6	0 3
Design 1 (0.53m, 73.2°, 84.2°)					
Reachability-Index	**0.660**	**0.591**	0.159	0.422	0.617
Design 2 (0.49m, 46.5°, 145.4°)					
Reachability-Index	0.069	0.054	**0.974**	0.110	0.328
Design 3 (0.53 m, 13.20°, 140.9°)					
Reachability-Index	0.118	0.180	0.581	**0.537**	0.739
Design 4 (0.51m, 64.03°, 61.49°)					
Reachability-Index	0.584	0.108	0.877	0.170	**0.888**

TABLE III: Four robot arm designs generated by MO-BBO. Best performers highlighted in bold. [Best viewed in color]

When we compute reachability index, we use local IK solvers that are sensitive to initial guesses. If bad initial guesses are used, all points on the sphere would be unreachable. On the other hand, globally IK solves are computationally costly. We choose to have the robot reach one end-effector position (seed position) via local IK and then propagate to nearby positions by using the IK solution as initial guesses. We randomly sample end effector poses within the grid defined in each scenario, and use a local IK solver to generate a configuration where the initial seed is sampled uniformly at random in the configuration space. We generate at most 100 samples for each design using this method. If there are no valid local IK solutions, we sample 5000 arm configurations uniformly at random and use them as the behavior variables.

After the initialization, we run 100 outer iterations with parameter settings $S = 20$ and $\kappa = 2$ were used. 56 designs lie on the the Pareto front, and Tab. III shows 4 selected designs produced by subsampling. These demonstrate a wide range of different performance characteristics, indicating that no design is ideal for all tasks.

VI. Conclusion & Future Work

We presented MO-BBO, a new algorithm for robot design optimization that is able to handle form and behavior co-design, multi-objective problems, and expensive performance measurement. Our main idea was to expand the Pareto front in a bilevel manner, in which the inner optimization chooses a design- and environment-specific decision. MO-BBO is demonstrated to effectively explore a Pareto front on two robot co-design problems.

In future work we plan to address more complex behavioral representations, such as policies or trajectories. Furthermore, an interesting remaining problem is how to scale co-design problems to very large numbers of environments where it would be prohibitively expensive to perform behavior optimization and performance measurement for every environment. Correlations between optimal behaviors and performance metrics in similar environments could be used to quickly eliminate unpromising designs. If environments themselves are parameterized, prior work in contextual GPs [14] might yield insights into methods for leveraging correlations between environments. Another approach might accelerate MO-BBO by allowing partial experiments. As pointed out in [22], contextual bandit algorithms allow certain experiments (e.g. neural network training) to be half-way done, and then selectively continue promising experiments.

REFERENCES

[1] A. Albers and J. Ottnad, "Integrated structural and controller optimization for lightweight robot design," in *2009 9th IEEE-RAS International Conference on Humanoid Robots*, 2009, pp. 93–98.

[2] R. Ariizumi, M. Tesch, K. Kato, H. Choset, and F. Matsuno, "Multi-objective optimization based on expensive robotic experiments under heteroscedastic noise," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 468–483, 2016.

[3] S. Belakaria, A. Deshwal, N. K. Jayakodi, and J. R. Doppa, "Uncertainty-aware search framework for multi-objective bayesian optimization." in *AAAI*, 2020, pp. 10 044–10 052.

[4] I. Char, Y. Chung, W. Neiswanger, K. Kandasamy, A. O. Nelson, M. Boyer, E. Kolemen, and J. Schneider, "Offline contextual bayesian optimization," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Curran Associates, Inc., 2019, pp. 4627–4638. [Online]. Available: http://papers.nips.cc/paper/8711-offline-contextual-bayesian-optimization.pdf

[5] T. Chen, M. Haas-Heger, and M. Ciocarlie, "Underactuated hand design using mechanically realizable manifolds," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7392–7398.

[6] N. Cheney, R. MacCurdy, J. Clune, and H. Lipson, "Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding," in *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, 2013, pp. 167–174.

[7] S. Doncieux, N. Bredeche, J.-B. Mouret, and A. E. G. Eiben, "Evolutionary robotics: What, why, and where to," *Frontiers in Robotics and AI*, vol. 2, p. 4, 2015. [Online]. Available: https://www.frontiersin.org/article/10.3389/frobt.2015.00004

[8] M. T. Emmerich, A. H. Deutz, and J. W. Klinkenberg, "Hypervolume-based expected improvement: Monotonicity properties and exact computation," in *2011 IEEE Congress of Evolutionary Computation (CEC)*. IEEE, 2011, pp. 2147–2154.

[9] E. Erkut, Y. Ülküsal, and O. Yeniçerioğlu, "A comparison of p-dispersion heuristics," *Computers & Operations Research*, vol. 21, no. 10, pp. 1103 – 1113, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0305054894900418

[10] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 7576–7586. [Online]. Available: http://papers.nips.cc/paper/7985-gpytorch-blackbox-matrix-matrix-gaussian-process-inference-with-gpu-acceleration.pdf

[11] D. Ha, "Reinforcement learning for improving agent design," *Artificial life*, vol. 25, no. 4, pp. 352–365, 2019.

[12] S. Ha, S. Coros, A. Alspach, J. M. Bern, J. Kim, and K. Yamane, "Computational design of robotic devices from high-level motion specifications," *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1240–1251, 2018.

[13] D. Hernández-Lobato, J. Hernandez-Lobato, A. Shah, and R. Adams, "Predictive entropy search for multi-objective bayesian optimization," in *International Conference on Machine Learning*, 2016, pp. 1492–1501.

[14] A. Krause and C. S. Ong, "Contextual gaussian process bandit optimization," in *Advances in neural information processing systems*, 2011, pp. 2447–2455.

[15] T. Liao, G. Wang, B. Yang, R. Lee, K. Pister, S. Levine, and R. Calandra, "Data-efficient learning of morphology and controller for a microrobot," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2488–2494.

[16] T. Lu, D. Pál, and M. Pál, "Contextual multi-armed bandits," in *Proceedings of the Thirteenth international conference on Artificial Intelligence and Statistics*, 2010, pp. 485–492.

[17] T. Morzadec, D. Marcha, and C. Duriez, "Toward shape optimization of soft robots," in *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE, 2019, pp. 521–526.

[18] R. Pinsler, P. Karkus, A. Kupcsik, D. Hsu, and W. S. Lee, "Factored contextual policy search with bayesian optimization," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7242–7248.

[19] C. Reyes and F. Gonzalez, "Mechanical design optimization of a walking robot leg using genetic algorithm," in *Climbing and Walking Robots*. Springer, 2005, pp. 275–284.

[20] R. Saravanan, S. Ramabalan, N. G. R. Ebenezer, and C. Dharmaraja, "Evolutionary multi criteria design optimization of robot grippers," *Applied Soft Computing*, vol. 9, no. 1, pp. 159 – 172, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494608000525

[21] N. Srinivas, A. Krause, S. M. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: No regret and experimental design," *arXiv preprint arXiv:0912.3995*, 2009.

[22] G. Sui and Y. Yu, "Bayesian contextual bandits for hyper parameter optimization," *IEEE Access*, vol. 8, pp. 42 971–42 979, 2020.

[23] K. Svanberg, "Mma and gcmma-two methods for nonlinear optimization," *vol*, vol. 1, pp. 1–15, 2007.

[24] M. Turchetta, A. Krause, and S. Trimpe, "Robust model-free reinforcement learning with multi-objective bayesian optimization," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 702–10 708.

[25] B. Yang, G. Wang, R. Calandra, D. Contreras, S. Levine, and K. Pister, "Learning flexible and reusable locomotion primitives for a microrobot," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1904–1911, 2018.

[26] F. Zacharias, C. Borst, and G. Hirzinger, "Capturing robot workspace structure: representing robot capabilities," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2007, pp. 3229–3236.

[27] H. Zhang, A. S. Kumar, J. Y. H. Fuh, and M. Y. Wang, "Design and development of a topology-optimized three-dimensional printed soft gripper," *Soft robotics*, vol. 5, no. 5, pp. 650–661, 2018.

[28] A. Zhao, J. Xu, M. Konaković Luković, J. Hughes, A. Speilberg, D. Rus, and W. Matusik, "Robogrammar: Graph grammar for terrain-optimized robot design," *ACM Transactions on Graphics (TOG)*, vol. 39, no. 6, pp. 1–16, 2020.