MULTI-OBJECTIVE BILEVEL BAYESIAN OPTIMIZATION FOR ROBOT AND BEHAVIOR CO-DESIGN

BY

YEONJU KIM

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Mechanical Engineering
in the Graduate College of the
University of Illinois Urbana-Champaign, 2021

Urbana, Illinois

Advisers:

Associate Professor Kris Hauser
Assistant Professor João Ramos

Abstract

Traditionally, the robot design process is based on the trial-and-error approach that involves repeated cycles of design, prototyping, and evaluation. During the process, the robot designer should tackle multiple objectives, which are commonly in conflicting relationships. Furthermore, the robot design assessment involves costly behavior optimization and performance evaluation in multiple environments. We propose a Multi-Objective Bilevel Bayesian optimization (MO-BBO) algorithm to automate the co-design process of the robot design and behavior simultaneously. Since the behavior should be optimized for each design and environment, we select the next design candidate in a bilevel manner. Design parameters and behavior parameters are the high- and low-level decision variables, respectively. We applied our algorithm to two robot co-design problems: gripper design problem and robot arm placement problem. MO-BBO was able to effectively expand the Pareto front in objective space on two problems.

We extend the robot arm placement problem by integrating human-likeness into objectives. To account for human likeness, we construct trajectory-based metrics to evaluate how well the robot arm follows the human motion trajectories extracted from the TUM Kitchen dataset and how similar the robot arm structure is to the human arm structure while following the trajectories. Compared to the designs generated by using reachability indices, the designs generated by the trajectory-based metrics have better performance when following human motion trajectories, especially in terms of collision rate and structural similarity.

Table of Contents

Chapter 1

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

1.1 Introduction

Robot design is difficult not only for novice designers but also for skilled engineers because a robot's performance is determined both by its design and its behavior. Hence, to completely understand the impact of a given design choice, at the design step, the designer must consider the robot's performance that is determined by behavior adapted to the different environments. It is important because the robot's best performance is regarded as its ”true” performance. Finding the optimal behavior for each design and each environment requires the designer to co-work with the robot behavior software engineers, and it makes the whole design process more time-consuming.

So design automation tools would be useful for designers as it can reduce the difficulty of designing a mechanical system as well as its behavior accordingly. Because classical design optimization methods cannot be applied on this problem directly, co-design of behavior and design for robots has been a topic of active research.

Besides the behavior optimization, the robot design process is known as complicated and time-consuming since the traditional design process is based on the trial-and-error approach. It takes repetitive design cycles, including ideation, design, prototyping, and testing. During the design step, the designer has to make a selection for numerous design parameters in a wide design space. Design parameters include a robot's morphology, kinematics, materials, and actuator parameters. Since the design parameters are correlated and sometimes counter-intuitive in terms of its effect, the relationship cannot be identified easily, which makes the designer hard to create a promising design in an early stage. During the test step, the designer either observes its performance in simulation environments or directly assesses its performance by prototyping and testing in the real world. A true assessment of its performance needs behavior optimization. In addition, designers have to define metrics used for evaluation by considering both its objectives and deploying environments. Furthermore, there exist the multiple objectives commonly in a conflicting relationship which cannot be optimized simultaneously. Also the priority among them is not clearly identified by the designer.

Prior works have addressed robot design problem computationally, but none of them cannot be directly applied to this problem while considering all the issues at the same time. For example, although evolutionary algorithms have been widely adopted to the robot design problem and there is well-established multi-objective optimization literature, it requires an immense number of experiments, which is not practical to the design problem with costly experiments. The gradient optimization method is used to optimize topology and controller parameters in a bilevel manner in [56]. But gradient based optimization cannot handle uncertainty and non-smoothness of the objective function. Contextual Bayesian Optimization proposed in [9] demonstrate the environment-aware objective function, which tackles multi-task problem. Even though this method allows to pick an optimal action for each task, each task is evaluated with a single reward function, which cannot be applied to co-design problems with multi-objectives for multi-environments. Bayesian optimization framework has been proposed to optimize both design and behavior in [32], but their method does not consider multi-environments or multi-objective settings.

We propose a Multi-Objective Bilevel Bayesian Optimization algorithm that can accommodate all the features of robot/behavior co-design problems by modeling the two-stage, three-way coupling between the robot's design, behavior, and the environments. In the BO framework, approximation of objective functions via Gaussian process(GP) provides posterior distribution of function values, which is much cheaper evaluation than the real observation. At each iteration, we select design parameters by maximizing the acquisition function constructed based on the GP posterior. The acquisition function is built to predict how the potential design can expand the Pareto front in the objective space. We propose the bilevel framework of the acquisition 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. After choosing a pair of design and behavior, it is evaluated on either the simulated system or the real system.

We employ this framework in two robot co-design problems: gripper design and arm placement for a bimanual mobile manipulator. The Pareto front is successfully expanded during the optimization and the designs on the pareto front have a wide range in each metric.

1.2 Literature Review

Codesign of robot design and behavior is a very demanding process so that skillful robotics engineers should spend a tremendous amount of time and manual efforts. There are several approaches to solve this problem computationally.

1.2.1 Local, Gradient Based Optimization

The method of moving asymptotes(MMA) [47], which is a gradient based optimization method, was used for solving different structural optimization problems. The topology of soft robotic finger is optimized in [56], resulting in the design that resembles human's finger. A. Albers et al. [1] integrated the optimization for controller parameters by simplex search algorithm (gradient-free algorithm) into the topology optimization using MMA in order to take into account the effect of coupling between the shape and controller.

Reinforcement learning(RL) has recently emerged as another way to learn a design and control policy simultaneously. RL takes a reward function that takes into account interaction with the physical environment. Schaff et al. [44] obtain optimal design parameters while jointly optimizing its corresponding policy by using Proximal Policy optimization. D. Ha [25] uses the policy gradient method, REINFORCEMENT, to solve co-design problem. These works attempt to learn the policy for each design candidate, which is the limitation that these suffer from the running time. Furthermore, those works employed policy gradient-based methods which is a form of local optimization.

Those methods yield only local improvement and cannot be easily generalized to other applications and environments.

1.2.2 Global Optimization

To address global optimization, evolutionary algorithm(EA) [15] has been widely used. This approach has been deployed in designing a soft robot [11], robot leg [40], and a gripper [42]. Further, several algorithms that extended the evolutionary algorithm to address multi-objective optimization have been proposed: NSGA-II [14], ParEGO [31] and MOEA/D [57]. Due to the difficulty of specifying preference among multiple objectives in robot design process, construction of a single objective is hard for the designer. Instead, providing optimal trade-offs may allow the designer to obtain useful insights and a wide range of choices. Hence, multi-objective evolutionary algorithms that generate trade-offs have been applied in designing a soft pneumatic robot [7], a mechatronic system [13], and a flexible spacecraft [22]. Although EA can effectively solve the non-convex and non-linear problem with continuous design space, it is not suitable when experimental costs are computationally expensive.

There have also been approaches for modular robot design where a set of modular components is designed to compose the robot design [26], [58], [52]. S. Ha et al. [26] used A* algorithm with a heuristic function to guide search in discrete design space. Zhao et al. [58] searched for the best robot design in discrete design space constrained by grammar. Recently, Whitman et al. [52] applied

Deep Q network to find optimal arrangement of modular designs. However, those approaches solve discrete optimizations and the designer should design a finite set of robot modules.

1.2.3 Bayesian Optimization

Bayesian Optimization(BO) is a global optimization method for black-box function with high evaluation costs, which can be applicable to the robotic system that is commonly expensive to run experiment. Hence, BO has been used to optimize mechanical design [36], the control policies [50], [38], and both of them simultaneously [32]. BO consists of steps for learning statistical models, called as *surrogate functions*, to approximate objective functions and determining designs to be evaluated next by optimizing the *acquisition function*. The statistical model, usually Gaussian process, provides a posterior probability distribution at a candidate that has not been evaluated yet. Based on that, the acquisition function measures how desirable the evaluation of the objective function with a specific design is expected to be for the optimization. By optimizing for the acquisition function, the next candidate is selected. Then we evaluate the chosen design, update the dataset, and fit the statistical models on the updated dataset.

To tackle variations of optimization problem such as multi-task, several researchers propose extended BO frameworks. For the multi task setting, action needs to be chosen based on the contextual information of environment. Swersky et al. proposed BO method to optimize a single action across all tasks [48]. Contextual Bayesian Optimization proposed in [9] is more suitable approach since it selects an optimal action for each task. Although those prior works take into account environment-aware decisions, the decision variables are not divided into design and behavior. Multi-objective BO was used in [2] to optimize the gait model of snake robot considering several challenges of robot design problems: multi-objective setting, expensive cost of observation in terms of time or money, and the noise of observation. Expected Improvement of Hypervolume(EIHV) [19] was used as an acquisition function. However, the analytic computation of hypervolume is complex and the authors did not consider environment-aware behavior. Liao et al. [32] adopted BO to design the legged robot and its behavior policy using a bilevel framework same as in our method. However, they only addressed the case of single-objective optimization.

To the best of our knowledge, there is none of method that incorporates all the properties of the co-optimization problem of robot design and environment-dependent behavior for the multiple objectives. We define the problem in general formulation in 1.3 and propose a novel method based on BO in 1.4.

1.3 Problem Formulation

Our optimization framework aims to generate trade-off designs in the Pareto front in the metric space. To use this framework, the robot designer should first define design parameters, behavior parameters, how to represent the performance into the numerical score, and environments where the design is tested with behavior. In our framework, we define the notations as below: design candidate d in design space D, behavior parameter θ in behavior space Θ, and a finite set of environments E. There are two kinds of performance measures: 1)*Design measures*, which depend on design parameters only, for example, the mass of the gripper. 2)*Behavior measures*, which measure the performance determined by design and behavior parameters in each environment. To illustrate, an example of behavior measure can be the grasp quality of the gripper design with control policy dependent on object shape, where the control policy is determined by behavior parameters and each object to be grasped indicate each environment. Thus, the behavior measure is a function of (design, behavior, environment). We consider design space D and behavior space Θ as continuous spaces. We assume that the designer provide N design measures $\{f_n(d)|n = 1, \dots, N\}$ and M behavior measures $\{g_m(d, \theta, e)|m = 1, \dots, M\}$.

1.3.1 Construction of Metric Space

If the robot problem with $M > 1$ behavior measures from a single performance evaluation, the behavior parameters should be optimized considering all of M metrics. To simplify the problem, we first assume that we have a single behavior measure.

The primary optimization takes place in design metric space, so the behavior measures need to be mapped into the design metrics which are functions of design. In this section, we introduce how to factor out the behavior and environment variables and construct the design metric space.

We can factor out the behavior variables first. Given a design $d \in D$, and an environment $e \in E$, the behavior parameters should be optimized as follows:

$$\theta^*_{d,e,m} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, g_m(d, \theta, e), \quad (m = 1) \tag{1.1}$$

After behavior is optimized, the behavior measure is evaluated as $g_m(d, \theta^*, e)$.

We can also factor our the environment variables. To demonstrate, we can take the grasp quality as an example again. Assume that we have 5 different objects to be grasped, which means $|E| = 5$. There may exist several use cases, but we consider two use cases here:

1. The designer may want to obtain the optimal solutions maximizing average performance over all environments. We can calculate the average metric as $\frac{1}{|E|} \sum_{e \in E} g_m(d, \theta^*_{d,e}, e)$.

2. The designer has some preference among environments or want to categorize the environment into some groups. For example, let the first two objects have cylindrical and the last three objects have box-shaped. Let's assume that the designer wants to construct the metric space as average over cylindrical objects and average over box-sahped objects. Then the environment set can be divided using a partition $\{B, C\}$, where B is a set of box-shaped objects and C is a set of cylindrical objects. Two design metrics are defined as $\frac{1}{2} \sum_{e \in C} g_m(d, \theta^*_{d,e}, e)$ and $\frac{1}{3} \sum_{e \in B} g_m(d, \theta^*_{d,e}, e)$.

To integrate those use cases into a common equation, we introduce a weighting matrix $W \in \mathbb{R}^{P \times |E|}$ where P is the number of elements in partition of environment set E. Thus, for the first case,

$$W = \begin{bmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{bmatrix}$$

and for the second case,

$$W = \begin{bmatrix} \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

Using W and optimized behaviors θ^*, we eventually represent the behavior-dependent metrics in (1.2) that are mapped into additional design measures.

$$f_{N+M(j-1)+m}(d) = W_j \begin{bmatrix} g_m(d, e_1, \theta^*_{d,e_1,m}) \\ \vdots \\ g_m(d, e_{|E|}, \theta^*_{d,e_{|E|},m} \end{bmatrix} \tag{1.2}$$

where W_j is the jth row of W. The dimension of metric space is $R = N + PM$, with the first N equivalent to N design measures and the last PM being behavior measures multiplied by the weighting matrix W.

Definition 1.3.1 (Design Metric)**.** The *design metric* $\mathbf{f} : D \rightarrow \mathbb{R}^{N+PM}$ maps a design to a *metric vector* $\mathbf{f}(d) = (f_1(d), \ldots, f_R(d))$.

To define Pareto optimal solutions in metric space, we have to unify whether each metric has to be maximized or minimized. We assume that the designer provides the metric equations, for which larger values are desirable. According to Pareto dominance relation, Pareto front is defined as follows:

Definition 1.3.2. 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 $\mathcal{S} \subset D$ 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* **PF** $= \{\mathbf{f}(d) | d \in D_{\mathbf{PO}}\}$

1.4 Multi-Objective Bilevel Bayesian Optimization(MO-BBO)

1.4.1 Multi-objective Bayesian Optimization(BO)

We adopt BO that is an effective framework to optimize expensive black-box functions that have no simple analytic equations as demonstrated in 1.2.3. For multi-objective optimization problems, there are so many variants of acquisition functions, aiming to numerically measure the possibility of each design to expand the Pareto front. The most intuitive acquisition function is the Expected Improvement of Hypervolume Increase(EIHV) [20], which corresponds to the expected gain of Hypervolume of evaluated designs in metric space. Hypervolume of Pareto front $\mathcal{H}(\mathbf{PF})$ is defined as the volume of the dominated subspace, bounded by a reference point $\mathbf{r}$ which is dominated by each element of $\mathbf{PF}$:

$$\mathcal{H}(\mathbf{PF}) = \text{Vol}(\{\mathbf{y} \in \mathbb{R}^R | \exists \mathbf{y}' \in s.t. \mathbf{y}' \succ \mathbf{y} \text{ and } \mathbf{y} \succ \mathbf{r}\}) \tag{1.3}$$

The hypervolume-based improvement is then defined as $I(\mathbf{y}, \mathbf{PF}) = \mathcal{H}(\mathbf{PF} \cup \{\mathbf{y}\}) - \mathcal{H}(\mathbf{PF})$. The expected value is computed by using the predictive distribution of the surrogate functions. The exact computation has been considered at most 2 dimensional space in [20]. For metric space dimension greater than 2, expensive computation at each point naturally leads to much expensive optimization of the acquisition function.

The other approaches based on uncertainty have recently been proposed. Uncertainty-aware search framework [6] first use acquisition function as a virtual objective to generate Pareto front and pick one design that maximizes uncertainty measure. Entropy based approach was proposed in [27], in which it aims to reduce the predictive entropy of the posterior distribution over the Pareto front.

We build a new acquisition function, Likelihood of Pareto Expansion (LPE), that is evaluated at each design. The computation of the acquisition function is accompanied by behavior optimization, which uses the surrogate functions approximating the performance measures attributed by design, behavior, and environment. Thus, the algorithm adopts the bilevel framework, in which the design effect is computed in the outer loop and the behavior is optimized in the inner loop.

1.4.2 Surrogate function: Gaussian Process

Gaussian Process(GP) is the most commonly used model for surrogate function. A GP is characterized by its mean function, $\mu(\cdot)$ and covariance function $k(\cdot,\cdot)$. Suppose we observe function values at $x_1,\ldots,x_k$. Given those observations using Bayes' rule, we can compute the conditional distribution of function values at points that have not been evaluated yet, called the posterior probability distribution. Conditional distribution is a simple Gaussian. This relatively cheap computation to predict function value at a new point is used to construct acquisition functions.

We use $(N + M|E|)$ GPs to approximate N design measures and M behavior measures for $|E|$ environments.

$$f_n(d) \sim \mathbf{GP}_n(d; \mathcal{B}_n) \tag{1.4}$$

$$g_m(d,e,\theta) \sim \mathbf{GP}_{m,e}([d,\theta]; \mathcal{B}_{m,e}) \tag{1.5}$$

Let's define the dataset $\mathcal{B}_* = \{(x^{(i)}, y_*^{(i)})|i = 1,\ldots,n\}$. $y_*(x) \sim \mathbf{GP}_*(x; B_*)$ means that y_* shows Gaussian distribution given the dataset $\mathcal{B}_*$. The conditional distribution is given by $y_* \sim \mathcal{N}(\mu_*(x), \sigma_*(x))$, where

$$\mu_*(x) = \mathbf{k}_{*,x}^T \mathbf{K}_{*,*}, \mathbf{y} \tag{1.6}$$

$$\sigma_*^2(x) = k(x,x) - \mathbf{k}_{*,x}^T \mathbf{K}_{*,*}^{-1} \mathbf{k}_{*,x} \tag{1.7}$$

and $\mathbf{y} = [y_*^{(1)},\ldots,y_*^{(n)}]^T$ is the set of observations, $\mathbf{K}_{*,*}$ is an $n \times n$ matrix of kernel values with ith row and jth column element being $k(x^{(i)}, x^{(j)})$, and $\mathbf{k}_{*,x}$ is a $n \times 1$ column matrix of kernel values evaluated between the prior input-output pairs $\mathcal{B}_*$ and x, with ith row element being $k(x^{(i)}, x)$. As iteration progresses, the dataset is updated, the GP is fitted taking $\mathcal{B}_*$, optimizes the covariance function parameters, and compute inverse matrix used in equation (1.7).

1.4.3 Acquisition function

We propose a new acquisition function, the Likelihood of Pareto Expansion(LPE) metric, which estimates the probability that it is non-dominated by the current Pareto front based on the surrogate function estimates. We measure the acquisition function value at design $d \in D$. First we have to estimate the metric values at d from the posterior Gaussian distribution. For the first N design metric, we can directly evaluate the mean $\mu_n(d)$ and variance $\sigma_n^2(d)$ of $\mathbf{GP}_n$. For the last PM behavior-dependent and environment-dependent metrics, we first compute the mean $\mu_{m,e}(d, \theta^*_{d,e,m})$ and variance $\sigma^2_{m,e}(d, \theta^*_{d,e,m})$ from $\mathbf{GP}_{m,e}$ at $[d, \theta^*_{d,e,m}]$. For $(N+M(j-1)+m)$th metric, a mean and variance can be represented as $W_j[\mu_{m,e_1}, \dots, \mu_{m,e_{|E|}}]^T$ and $W_j[\sigma^2_{m,e_1}, \dots, \sigma^2_{m,e_{|E|}}]^T$, respectively. Overall, metric space vector $\mathbf{f}$ takes on a Gaussian distribution $\mathbf{f} \sim \mathcal{N}(\mu, \Sigma)$. From the distribution, we draw K samples $\{\mathbf{f}^{(k)}(d) | k = 1, \dots, K\}$. We define LPE metric at d, $\alpha(d)$, is the ratio of the number of non-dominated samples by the current Pareto front:

$$\alpha(d) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}[\mathbf{f}^{(k)}(d) \succ \mathbf{PF}]. \tag{1.8}$$

As the computation of the acquisition function includes the optimization of the behavior parameters, the way to optimize behavior for each environment and each design needs to be defined. Inspired by upper-confidence based algorithm analyzed in [45], we optimize behavior for the upper bound of a confidence interval of $\mathbf{GP}_{m,c}$ as follows:

$$\theta^*_{d,e,m} = \underset{\theta}{\operatorname{argmax}}\, \mu_{m,e}(d, \theta) + \kappa \sigma_{m,e}(d, \theta), \tag{1.9}$$

where $\kappa \geq 0$ trades off exploration and exploitation. Large κ value leads to more exploration, while small κ encourage more exploitation. Equation (1.9) is a surrogate function optimization of equation (1.1), which is used for the construction of metric space to compute acquisition function. The performance are highly dependent on the selection of behavior parameters. In other words,

Algorithm 1 MO-BBO

```
Require: Iteration count $T$, initial data set $\mathcal{B}_n, \mathcal{B}_{m,e}$
Require: Upper-level sample size $S$, LPE sample size $K$
Require: Exploration-exploitation trade off $\kappa$
 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 Equation (1.9)
 7:     Compute $\mu(d)$, $\Sigma(d)$ from GPs and $\theta^*_{d,e,m}$
 8:     $d_t \leftarrow \mathrm{argmax}_{d \in D_t}\, \alpha(d)$ (Equation (1.8))
 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 PF
```

suboptimal decisions generated by the algorithm cause underestimation of the true performance of the design, which determines the quality of the Pareto optimal solutions.

In our work, maximizing of the acquisition function over design candidates become the following bilevel optimization problem.

$$\underset{d, \theta^*_{d,e,m}}{\mathrm{argmax}}\, \alpha(d), \tag{1.10}$$

$$\text{where } \theta^*_{d,e,m} = \underset{\theta}{\mathrm{argmax}}\, \mu_{m,e}(d, \theta) + \kappa \sigma_{m,e}(d, \theta) \tag{1.11}$$

1.4.4 Algorithm Pipeline

In this section, the MO-BBO (1) algorithm is illustrated in more detail. We first draw a set of random designs and behavior samples and evaluate the metric values. The Gaussian Processes are fitted on the dataset and the initial Pareto front is identified before the outer loop (lines 1-2). For each outer iteration, we find the design candidate and corresponding behavior to be evaluated. The design and behavior are chosen by globally optimizing Equation (1.10). Since we have to budget in terms of the computational cost spent on the optimization of the acquisition function, we approximate the solution by restricting the design space to a set of S designs sampled from the design space. Then in the low-level optimization (lines 5-6), for each design, we optimize for $\theta^*_{d,e,m}$ using an optimization of Equation (1.9). To optimize for behavior, we use two different methods and compare them: brute force uniform random sampling, and a gradient-free global optimization method (DIRECT). The number of function evaluations conducted during the low-level optimization is $fmax$. After choosing the design and behavior, the remaining steps (lines 9-13) are similar to a standard Bayesian Optimization framework: evaluate performance measures, update dataset $\mathcal{B}_*$, and the Gaussian processes are fitted on the updated dataset. Then from the observed performance measures, we compute the design metric vector and update design metrics as well as the Pareto front.

1.4.5 Performance Discussion

We analyze the computational cost of Algorithm 1. At the tth iteration, the cost of evaluating a surrogate function at some query point is $O(t)$. In the low-level optimization, let's assume J is the cost of solving global optimization of Equation (1.9) using DIRECT or uniform sampling. It should be done for all S designs and all $M|E|$ behavior measures so that a cost results in $O(SM|E|J)$. When DIRECT method is used with large sample size for the behavior parameters, this cost tends to dominate complexity. Using the optimal behavior $\theta^*_{d,e,m}$ for each design, the cost of evaluating the acquisition function is $O((N + M|E|)Kt)$ since K samples are compared with the design metric vectors in the Pareto front with size of $O(t)$. Re-fitting a surrogate function costs $O(t^3)$. Let's assume that C is the cost of evaluating each metric, which is high when evaluating the behavior measures in general. Overall, we have running time for each iteration as $O((N + M|E|)(KtS + C + t^3) + SM|E|J)$. And the total running time for T iterations, we have $O((N + M|E|)T(KTS + C + T^3) + TSM|E|J)$.

1.4.6 Post-Processing: Subsampling

If the designer deals with high-dimensional design metric space, the resulting Pareto front may contain a large number of Pareto optimal design. For example, in our experiment, 100 designs are used for initialization and 100 designs are generated by algorithm. The resulting Pareto optimal designs are 56 designs, which are not easy to be investigated. To provide a subset with manageable size, we propose a subsampling step after the design generation by the algorithm. The goal of subsampling is to create a subset with a manageable size which has uniform distribution in the metric space.

To solve this problem, we are inspired by the p-dispersion problem: Select p out of n given points $(1 < p < n)$ in some space, where the objective is to maximize the minimum distance between any two of the selected points. We choose one heuristic, called Greedy construction heuristic, among several heuristics that are used to solve the p-dispersion problem [21]. This heuristic starts by choosing the two furthest points in the Pareto front. The distance between two points are defined as a weighted Euclidean distance between the metric vectors, with each dimension normalized to the range [0, 1]. In each of the next $(p - 2)$ iterations, a new point is chosen to maximize the minimum distance to the points already in the subset. The distance between a certain point $\mathbf{f}$ and the points in the subset V is calculated as: $\min_{\mathbf{f}_i \in V} d(\mathbf{f}, \mathbf{f}_i)$.

1.5 Experiments

We applied our algorithm to two robot co-design problems: Gripper design problem 1.5.1 and robot arm placement problem 1.5.2.

1.5.1 Gripper Design Problem

The goal of the gripper design problem is to obtain a design that is light as well as can grasp various objects. We can expect that these two metrics may be in a conflicting relationship, since two-finger gripper and gripper with too short may not achieve a stable grasp.

The gripper design is basically composed of a finger with three links and a cylindrical palm. The design parameters include finger shape and the number of fingers. Finger shape is specified by the curvature of the curve of cross-section and the link length. The range of design space are defined

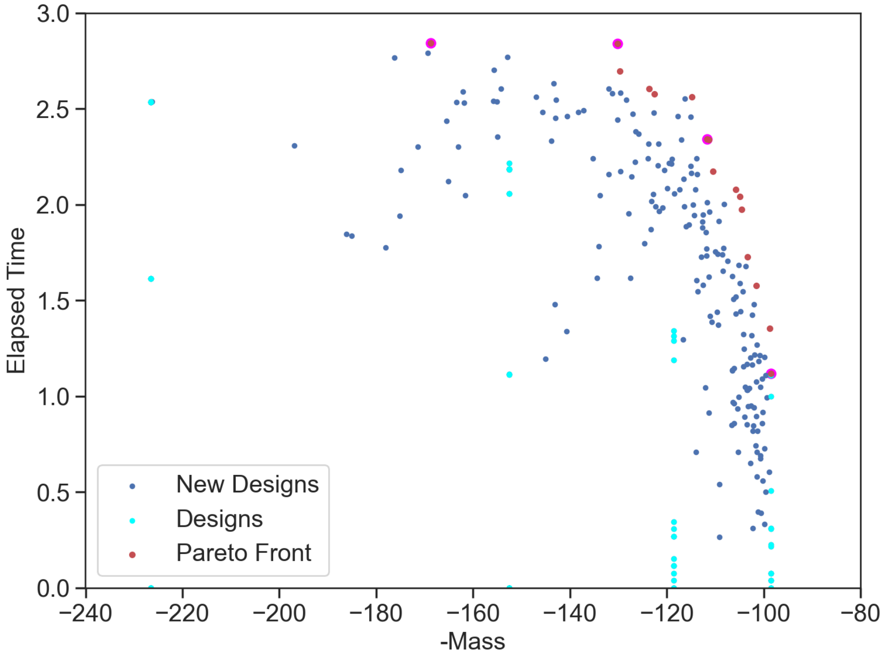


Figure 1.1: Metric values of 200 gripper designs explored by MO-BBO, of which 16 designs are on the Pareto front. The points colored in magenta are 4 subselected designs from the Pareto front.

by finger length (range [0.1m, 0.4m]), the number of fingers (2, 3, or 4), and finger curvature (range [-3, 3] in m^{-1}). The three exemplary designs are illustrated in Figure 1.2. The behavior space is defined by the approach direction and the gripper's rotation along z-axis. It is used for when the grasping simulation starts. And we constructed 13 objects for an object set by combination the geometric primitives to diversify the difficulty of grasping as shown in Figure 1.2 (d).

To evaluate the grasping ability of each design and behavior, we use grasping simulation including steps of approach, close, lift, and shake. The simulation starts with the gripper approaching the center of the object from a distance 3m. The behavior parameters determine the starting orientation and approach direction. The approach direction is defined using spherical coordinate $(-\cos\theta\cos\phi, -\sin\theta\cos\phi, -\sin\phi)$. ϕ and θ have ranges $[\frac{\pi}{4}, 0.99\frac{\pi}{2}]$ and $[0, 2\pi]$, respectively. This approach direction indicates the z-axis of the cylindrical palm which is the normal vector of a circular plane. And the z-axis rotation angle $\beta \in [0, \pi]$, which is not changed during the approaching step. The gripper approaches the object until it touches the ground or the object. If it touches the ground first, not the object, we regard it as a failed attempt. Otherwise, the gripper closes its fingers

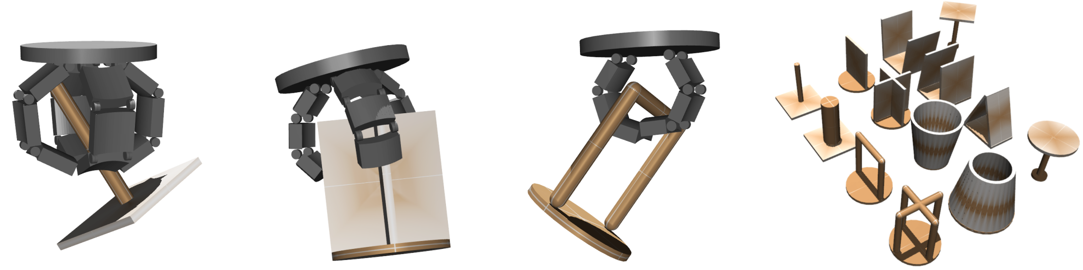

(a) (0.30, 4, -1.23) (b) (0.18, 3, 1.92) (c) (0.20, 2, -1.34) (d) Object set

Figure 1.2: Gripper design problem, with (a–c) showing example designs with parameters (finger length (m), the number of fingers, finger curvature (m^{-1})).

to grab the object. We regard the simulated grasp as a success if more than three links continuously make contact with the object for $0.1s$ or the max velocity of the joint becomes less than the threshold $5e^{-1}$. The way of counting the contacts between the object and the gripper is inspired by [29]. If the gripper succeeds to grab the object, it lifts the object vertically for $0.5s$. After lifting, it starts shaking along a sinusoidal wave with amplitude 0.2m and period $0.5s$ for 5 periods.

We define one design measure and one behavior measure for the performance measures. The design measure is $f_1(d)$ = Gripper-Mass$(d)^{-1}$. We take the reciprocal of the gripper mass to make the metric to be maximized. The behavior measure is the elapsed time metric $g_1(d, \theta, e)$ which measures the time elapsed from the lifting until the gripper drops the object and the object makes contact with the ground. To observe the grasp quality averaged over all objects, we use $W = \begin{bmatrix} \frac{1}{|E|} & \dots & \frac{1}{|E|} \end{bmatrix}$, $|E| = 13$ in our experiment. $(N + M|E|) = 1 + 1 \times 13 = 14$ Gaussian processes are used during the optimization. By eliminating the behavior and environment variables, design metrics are constructed as below and the metric space is 2-dimensional:

$$f_1(d) = \text{Gripper-Mass}^{-1}(d) \text{ and} \tag{1.12}$$

$$f_2(d) = \frac{1}{|E|} \sum_{e \in E} g_1(d, e, \theta^*_{d,e,1}) = \frac{1}{|E|} \sum_{e \in E} \text{Elapsed-Time}(d, e, \theta^*_{d,e,1}). \tag{1.13}$$

We run the optimization with parameter settings $(S, fmax, \kappa) = (20, 1000, 2.0)$ for $T = 200$ iterations. The metric space explored is shown in Figure 1.1, where we can see most of designs are generated near the Pareto front. Table 1.1 illustrates 4 selected designs from the Pareto front by

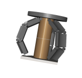
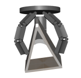
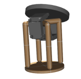
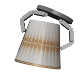
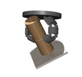
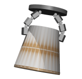
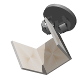
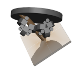
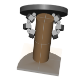
Object	1	2	3	4	5	6	7	8	9	10	11	12	13
Design 1 Mass 168.8													
Elapsed-Time	2.274	3.0	2.908	3.0	2.347	3.0	3.0	3.0	3.0	3.0	3.0	3.0	3.0
Design 2 Mass 130.2													
Elapsed-Time	2.521	3.0	3.0	3.0	3.0	3.0	2.427	1.641	3.0	3.0	3.0	3.0	3.0
Design 3 Mass 111.7													
Elapsed-Time	0.5	3.0	2.9	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 1.1: Four gripper designs from the Pareto Front of Figure 1.1. Images captured during lifting and shaking. Elapsed-Time indicates how long the object stays held during shaking, in seconds (max 3 s).

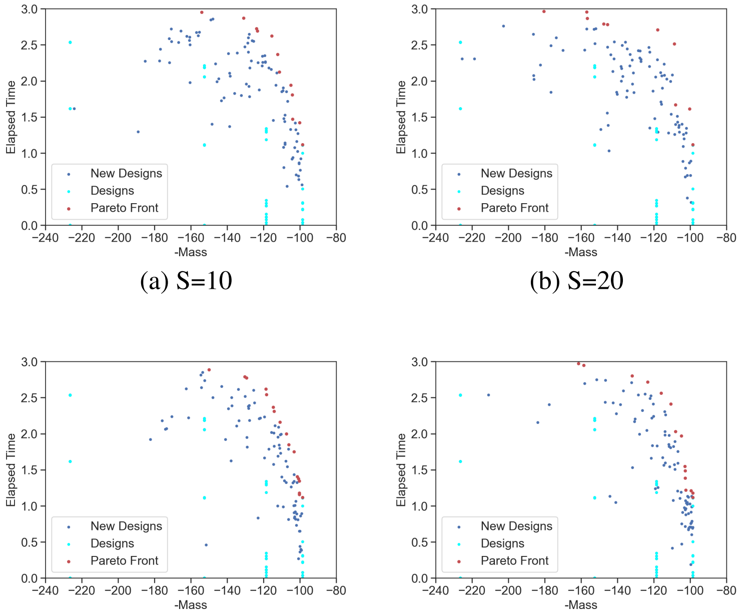
Metrics	Baseline	Sample size (S)			Confidence Bound (κ)				#func. evals. ($\max f$)		
	$(50, 2.0, 1k)$	100	20	10	0.0	1.0	5.0	10.0	10000	100	10
					Inner Optimization via Uniform sampling / DIRECT						
HV	2.703/2.746	2.646/2.693	2.684/2.759	2.713/**2.776**	**2.784/2.871**	2.753/**2.787**	2.580/2.706	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 1.2: 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.

using the subsampling algorithm mentioned in section 1.4.6.

Effect of Parameter settings Since the gripper design problem is not a high-dimensional problem and the time spent on experiments can be reduced by parallelization, we were able to analyze the effect of parameters used in the algorithm. To compare several parameter settings, the performance of algorithm with each setting should be evaluated by measuring the quality of the Pareto front. C. Audet et al. [3] reviewed several performance indicators used in multi-objective optimization. We adopted two measures: Hypervolume(HV) and Metric Spread(MS). Hypervolume is calculated using Equation (1.3). And, to examine the extents of the spread achieved in the Pareto front, we define the Metric Spread as the gap between the min and max value of points in the Pareto front. It is computed for each metric dimension as follows:

$$\max_{d \in D_{\mathbf{PO}}} f_i(d) - \min_{d \in D_{\mathbf{PO}}} f_i(d), \text{ for } i \in \{1, \dots R\} \tag{1.14}$$



(a) S=10

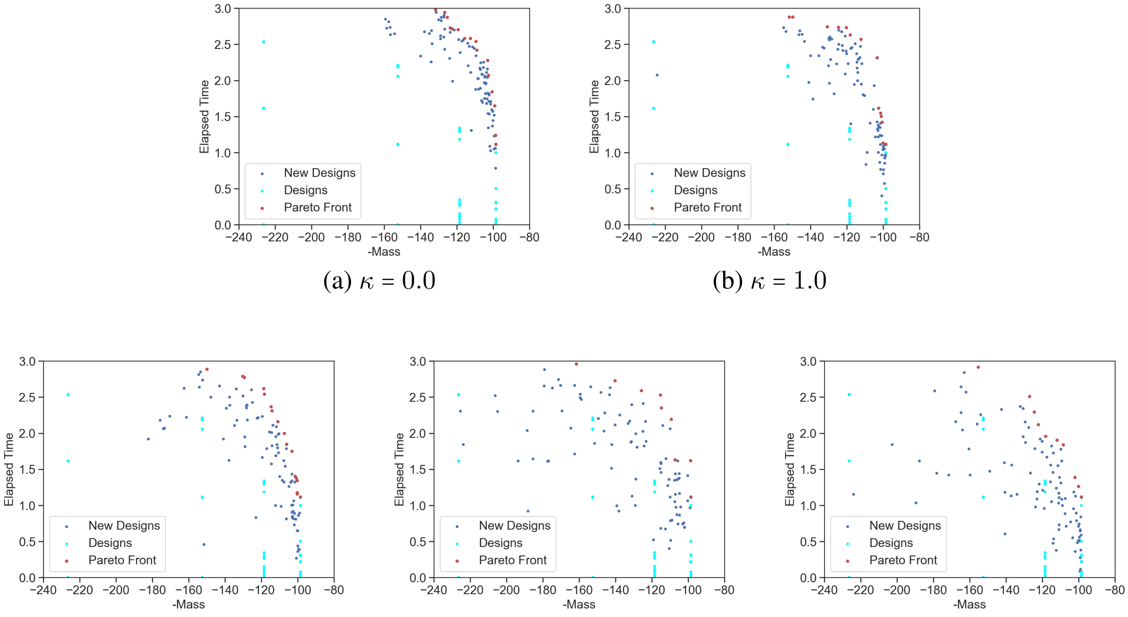
(b) S=20

(c) S=50 (Baseline)

(d) S=100

Figure 1.3: The Pareto front by changing the design sample size: 10, 20, 50, and 100.

For both metrics, the larger value means better performance. We observed the effect of parameter settings by changing the parameter value from the baseline $(S, \kappa, fmax) = (50, 2.0, 1000)$. To reduce the randomness, we conducts 4 trials for each setting, and take average over them. Table 1.2 shows the result. As design sample size S decreases, MS and HV are increased as shown in Figure 1.3. As κ increases, MS for the mass metric increases, but MS for the elapsed time metric and HV decreases as shown in Figure 1.4. Those results can be interpreted as the smaller value of κ and larger design sample size lead to decreasing the randomness when selecting the design candidate so that the design generation is more focused on the smaller mass region more. $fmax$ does not affect the performance significantly as much as the other parameters as shwon in Figure 1.5. But larger value in $fmax$ make running time of optimization longer. For example, in the case of $fmax = 10000$, it takes more than 10 hours to complete 100 designs generation. We also compared two different ways for the behavior optimization. For many cases, DIRECT works slightly better than the brute-force uniform sampling method. but DIRECT takes longer computation time (2.97 hours vs 1.66 hours).



(a) κ = 0.0

(b) κ = 1.0

(c) κ = 2.0(Baseline)

(d) κ = 5.0

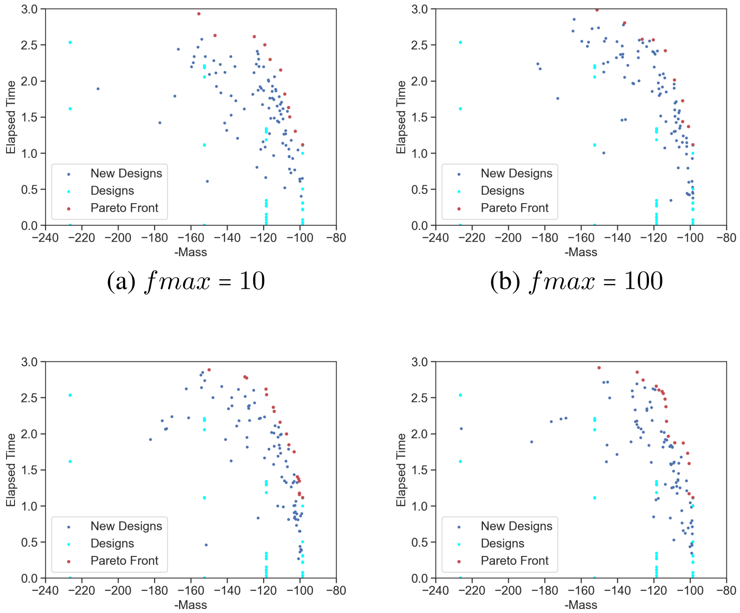
(e) κ = 10.0

Figure 1.4: The Pareto front by changing $kappa$: 0.0, 1.0, 2.0, 5.0, and 10.0.

1.5.2 Robot Arm Placement

The second application on co-design problem is robot arm placement problem. We aim to find the optimal robot arm placement that maximizes the reachability in human environments. The mounting orientation and shoulder width affect the capability of robot arms a lot, so we have to carefully choose them. Inspired by the reachability map analysis [53], we simplify reachability index and use that to construct the performance measure.

The three design parameters include pan angle(range $[0°, 90°]$), tilt angle(range $[0°, 180°]$), and the shoulder width(range [0.4 m, 0.65 m]). Three exemplary designs are illustrated in Figure 2.1. The behavior parameters consist of 6 joint angles of the left arm, which will be used as the Inverse Kinematics(IK) seed for reachability evaluation. Environments are defined picking up scenarios for ground and table and manipulation scenarios for low-shelf, high-shelf, and table. The scenearios are illustrated in Figure 2.1. In each scenario, a bounding box workspace is defined and a discretized grid of position targets with resolution 0.05m are defined in the bounding box as shown in Figure 2.1. We also define goal orientations according to the scenario. For the Ground and Table-vertical



(a) $fmax$ = 10

(b) $fmax$ = 100

(c) $fmax$ = 1000(Baseline)

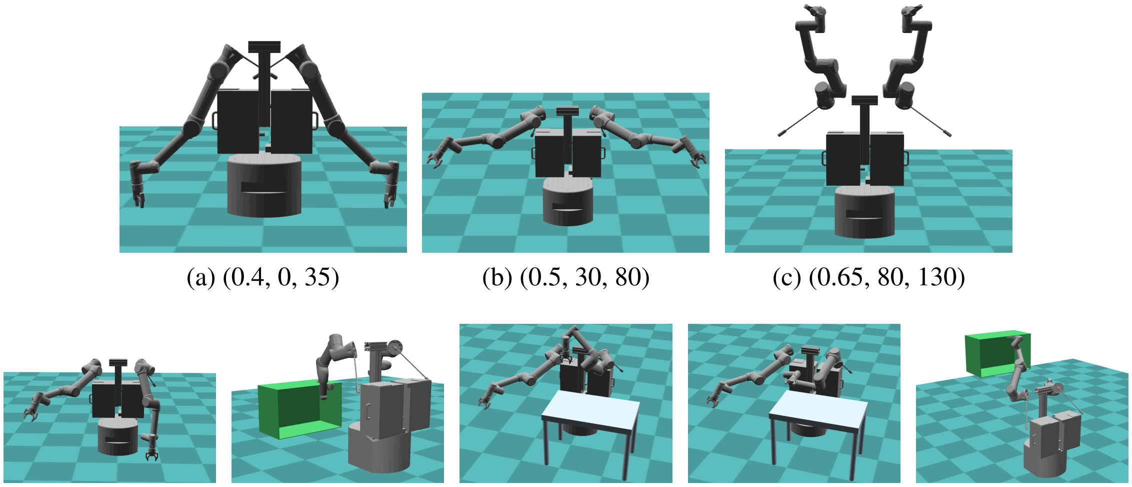
(d) $fmax$ = 10000

Figure 1.5: The Pareto front by changing $fmax$: 10, 100, 1000, and 10000.

scenario, a vertical orientation is only tested. In Table-horizontal scenario, we check 6 horizontal directions with yaw angles: $\{0°, 60°, 120°, 180°, 240°, 320°\}$. For the High-shelf and Low-shelf scenarios, we consider $\{-60°, 0°, 60°\}$ yaw angles.

We use average reachability index over every target positions as a behavior measure $g_1(d, \theta, e)$, which represents how the fraction of the workspace is reachable. On each target position, the reachability index is defined as the fraction of reachable orientations. For more details, 6DOF target pose is specified for each goal orientation, then we solve IK by using a local optimization (Newton-Raphson) method. We use a local optimization since reachability map constructed by solving IK globally in [53] is computationally costly to be evaluated for many designs. However the result from local IK solver is easily affected by the initial guess. Hence, we designate initial guess as behavior parameters to be optimized.

To choose initial guess efficiently, we generate valid samples of behavior parameters at the low-level optimization. The valid sample indicates 6-d configuration that makes the end-effector position within the workspace for each scenario. To generate samples, we randomly sample end-effector poses within the grid positions, and use a local IK solver to generate a 6-d configuration



(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

Figure 1.6: Robot arm design problem. (a–c) show 3 illustrative designs with design parameters (shoulder width (m), pan (°), tilt (°)). (d–h) show test scenarios.

where the initial seed is sampled uniformly at random in the configuration space. We generate at most 100 samples for each design. Resulting configurations are assigned to the behavior samples to be used for the behavior optimization in Equation (1.9). If there are no valid configurations found, we sample 5000 arm configurations uniformly at random and use them as the behavior parameter samples.

In this problem, $(N + M|E|) = 0 + 1 \times 5 = 5$ Gaussian processes are used to approximate the behavior measures. To inspect all behavior metrics, behavior measures are mapped into 5 dimensional metric space by assigning $W = I_{5\times5}$.

$$f_i(d) = g_1(d, e_i, \theta^*_{d,e_i,1}) = \text{Avg-Reachability}(d, e_i, \theta^*_{d,e_i,1}), \qquad \text{for } i = 1, \ldots, 5. \tag{1.15}$$

For the initialization, we evaluate 100 designs with valid samples. Then we run 100 iterations with parameters $(S, \kappa) = (20, 2.0)$. $fmax$ is replaced by 100 or 5000 which are used for valid sampling method. As a result, 56 designs lie on the Pareto front. A large number of Pareto optimal designs mean that there is no ideal design that has maximum values for every metric. Table 1.3 shows 4 designs selected by using subsampling algorithm. Each design and behavior pair have

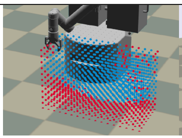
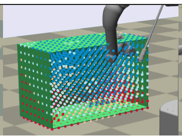
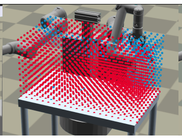
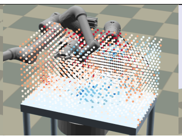
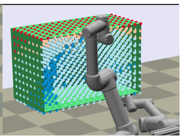
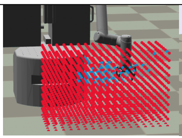
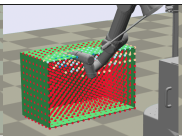
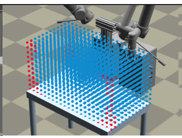
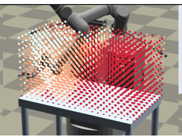
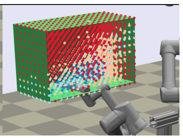
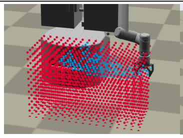
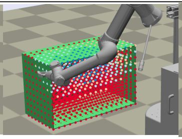
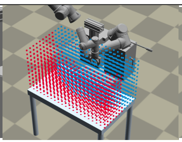
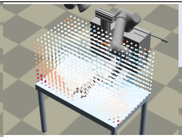
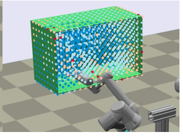
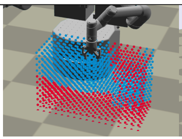
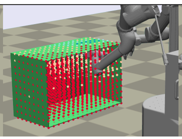
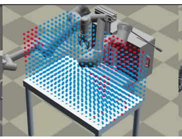
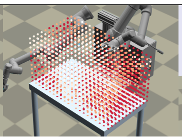
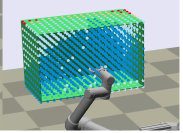
Scenario	Ground	Low-shelf	Table-vert	Table-horiz	High-shelf
Legend (# of target orientation reached)	0 🟥🟦 1	0 🟥🟧🟦🟦 3	0 🟥🟦 1	0 🟥🟧⬜⬜🟦🟦 6	0 🟥🟧🟦🟦 3
Design 1 (0.53m, 73.2°, 84.2°)					
Reachability-Index	**0.660**	**0.591**	0.159	0.422	0.617
Design 2 (0.49m, 46.5°, 145.4°)					
Reachability-Index	0.069	0.054	**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 1.3: Four robot arm designs generated by MO-BBO. Best performers highlighted in bold. [Best viewed in color]

good performance in a few metrics, not for all metrics.

1.6 Conclusion

In this chapter, we proposed MO-BBO, a new algorithm for robot design and behavior co-design problem having multi-objective and multi-environment settings. We constructed a new acquisition function used in this algorithm, the Likelihood of Pareto Expansion metric. To address the design- and environment-dependence of the robot behavior, the acquisition function is optimized in a bilevel manner, where inner optimization optimizes the behavior for all environments. We applied our algorithm to solve two robot co-design problem. During the optimization, design space and behavior space are effectively explored resulting in the Pareto front with a wide range.

In the future work, we can apply our algorithm to problems with more complex behavioral representations. One possible approach is to employ Model-based Reinforcement learning such as [12] to learn complex controller. Furthermore, we can scale up the problem to very large numbers of environments. Behavior optimization and performance measurement for all environments would be very expensive. As used in prior works [48], [9], if we were able to parameterize the environments, we efficiently learn a similar unknown environment using the data from previous experiments. Another approach might be allowing partial promising experiments as like the dynamic allocation method of training resources is integrated into contextual bandit algorithm in [46].

An interesting question of robot design optimization can be how to integrate users as a part of the optimization process and allow users to suggest the direction of search, instead of providing solutions only after optimization is complete. The interactive method could assist users in learning about complicated robot design problems. To make the optimization more interactive for users, there are several ways to articulate user opinions or preferences to be used during the optimization. For example, by specifying the region of interest in objective space, the user can guide the optimization to focus on that region proposed in [59], [43]. Also, there might be a use case where they do not satisfy the obtained designs or metrics since the metric definition in the numerical equation cannot represent the robot's performance well. In this case, we will be able to employ Preference Learning techniques to obtain user preference over the designs and use it as a guide during the optimization.

Furthermore, the visualization can be improved to show generated designs to users effectively. As the design space and the objective space are high-dimensional in general, it is hard to demonstrate the results with visualization. That is why we proposed the subsampling algorithm to extract a subset with a manageable size. The subsampling method could be improved by considering distance in design space as well as the distance in metric space.

Chapter 2

Robot arm placement of dual-arm robot used in human environments

2.1 Introduction

TRINA (Tele-Robotic Intelligent Nursing Assistant) is the robot platform being developed by Team AVATRINA. The operator controls the robot through off-the-shelf VR system, and the robot interacts with the humans in the remote environments. In the design process, two UR5 robot arms should be placed on the torso. We propose a new method for the robot arm placement optimization to satisfy our design goals 1) human-like movement 2) good manipulation performance.

Robot arm base placement affects the manipulation capabilities a lot, such as reachability, manipulability, and redundancy. So there have been various approaches to optimize the best base placement for different kinds of robotic systems. Workspace volume is used as the most important metric to be optimized in [5], [4], [17]. The reachability map proposed in [53] is also used to select the base placement in [55], [39]. For the bimanual system, the common reachability map and the cooperative reachability map are proposed and used as metrics in [10]. However, those indices lack consideration for the collision and continuous motion. Furthermore, since TRINA has a mobile base and will be used in human environments like hospitals and offices, the analysis of the whole reachable area is unnecessary. The performance in a compact region is a more important factor rather than the larger workspace volume. It is further supported by the observation that the locations of the wrists during daily tasks are concentrated in a compact region close to the human torso [28].

Besides the capability of robot arm, the human-likeness is a key factor to attain safety and social connection. When the robot conducts human-like robot motion, the human can predict robot's behavior easily and execute intuitive interaction with the robot. Also the human-likeness of robot contributes to increased likeability which facilitates the social connection.

Thus, our goal is to place the robot arm UR5 to the torso by maximizing human-likeness in the design process as well as the capability to conduct everyday tasks in the human environment. But the existing approaches applying human-likeness have focused on the motion planning and human to robot motion mapping problems. For example, motion generation algorithms using human likeness evaluation metrics are proposed in [54], [23]. And human motion is mapped to the robot reducing as much structural difference as possible in [33].

In this chapter we present an optimization method for robot arm placement. We propose novel metrics that evaluate how well the robot follows the human motion trajectories as well as how similar the structure of the robot arm is to the human arm structure during the motion. We extract the trajectories from TUM Kitchen Data Set [49] which is human pose data tracked when performing everyday activities in a kitchen environment. Based on the trajectory-based metrics we construct, the designs optimized using trajectory-based metrics are compared to the designs optimized using reachability indices.

2.2 Realated Work

2.2.1 Robot arm placement optimization

M. Bagheri et al. identified a range of upward and forward angle of shoulder base arrangement by optimizing several performance indices(workspace, manipulability, redundance, and Inertia Ellipsoid) [5]. They optimized performance metrics in predefined prioritized order, considering a single objective at once rather than using multi-objective optimization. These indices were also used for the development of Bi-manual robot in [4]. Based on [5], J. Duan et al. added dynamic performance indices and identified the optimal base frame placement [17].

Since the manipulability index calculated at a certain configuration is local measurement, the index cannot assess the general capability of robot arms. Also the workspace volume analysis specifies only the reachable positions, which does not consider information of orientations. Instead, Reachability Index and Capability map [53] have been utilized to find the optimal robot placement [35], [51] and robot arm mounting position in [10], [39]. We choose to use the Reachability Index as a baseline metric to compare with our optimization results.

2.2.2 Applications of Human-Likeness

Anthropomorphism is essential to attain safety and social connection through robot likeability [16], [18]. The human users prefer robots moving like humans as they feel more comfortable and safe interacting with them. To facilitate human-robot interaction and collaboration, the human-likeness should be considered. Human-likeness evaluation has been applied to generate human-like motion. M. J. Gielniak et al. generates human-like motions for social robots [24]. F. Zacharias et al. used Rapid Upper Limb Assessment(RULA), the criterion evaluates how ergonomically good robot's posture is, for the planning problem in [54]. For the teleoperation, the human-likeness has been taken into account to map human motion capture data to the robot arm motion. M. V. Liarokapis et al. [33], [34] proposed metrics that quantify functional anthropomorphism: Volume of the convex hull, distance between joint positions and human elbow, and area of triangles. The position and orientation goals is combined with the metric for anthropomorphism into a composite objective function, a form of weighted sum. Then by optimizing the objective function, they showed much more anthropomorphic motion is generated rather than using joint-to-joint mapping and inverse kinematics mapping.

To find robot arm position and orientation that makes them behave most similarly to humans, we quantify the similarity to human motion and apply it in the design process. We utilized hand, elbow, and shoulder position to assess the dissimilarity between the human and robot configurations. The metrics are described in section 2.3.2.

2.3 Method

2.3.1 Baseline: Reachability based metrics

Reachability Map

The reachability map represents the poses that can be reached in the Cartesian workspace of a robot arm. The workspace cube is defined and discretized into equally sized smaller cubes. Sphere is inscribed into each subcube with the diameter equal to the width of the subcube. Then N_p evenly distributed points are generated on the sphere according to the spiral point algorithm proposed by Saff et al. [41]. The x-axis is pointing towards its center and the y-axis and z-axis are tangentional to the sphere surface. The frame is rotated around its x-axis according to a fixed stepsize. Thus, N_r frames for each point on the sphere are defined. Each resulting frame is considered as a goal pose and the inverse kinematics(IK) solution is computed. If IK solution exists for one of the rotated frames at a point on the sphere, that point is marked as reachable.

$$D = \frac{R}{N_p} \cdot 100 \quad \text{with } R \leq N \tag{2.1}$$

Howard et al. [28] showed that the majority of elbow and wrist positions lies in front of a human body during natural everyday movements. Thus, we set frontal area as an area where we investigate for the reachability rather than analyzing the whole workspace of the robot arm proposed in [53]. The workspace cube is defined as x range [0.4m, 1m], y range [-0.5m, 0.5m], and z range [0.5m, 1.8m]. The subcubes are generated with the resolution 0.1m. The number of approach directions N_p on the sphere is 20. The stepsize for the rotation along x-axis is $60°$, then N_r is 6.

For each pose, we solve IK with 10 random restart. Then the collision is checked for terrain and other robot links except for the opposite arm as we need to calculate the reachability of each arm regardless the joint configuration of the opposite arm.

Metric Design

In [39], the mounting pose of robot arm is chosen by comparing workspace volume, high dexterity volume (above 75 percent reachability), and path feasibility around the storage areas. The high dexterity volume is represented in the equation

$$\text{High dexterity volume} = \frac{\text{Workspace of the dual arm robot with RI over 75}}{\text{Workspace Cube}} \tag{2.2}$$

C. Chen et al. [10] proposed two metrics for the robot arm placement problem using reachability analysis: Effective common area and Effective cooperative area. The common reachability is the geometric mean of the reachability index of its two arms, as shown in equation (2.3). Cooperative reachability measures if their end-effector can point to each other, usually required for assembly tasks. For a point on the sphere that can be reached by the left arm, we checked if a symmetric point can be reached by the right arm. If the symmetric point is reachable, it is marked as the reachable point on the sphere in terms of cooperative reachability. Similar to their metrics, we then defined the effective common area and the effective cooperative area in equation (2.5) and (2.4), respectively. The global workspace covers an area where each arm can reach, respectively.

$$\begin{array}{c}\text{Common RI in}\\\text{subregion X}\\\text{of dual arm robot}\end{array} = \sqrt{\begin{array}{c}\text{RI of left arm}\\\text{mapped into X}\end{array} \times \begin{array}{c}\text{RI of right arm}\\\text{mapped into X}\end{array}} \tag{2.3}$$

$$\begin{array}{c}\text{Effective common area}\\\text{of dual arm robot}\end{array} = \frac{\begin{array}{c}\text{Common workspace of the dual arm robot}\\\text{with common RI over 55}\end{array}}{\begin{array}{c}\text{Global workspace of}\\\text{a dual arm robot}\end{array}} \tag{2.4}$$

$$\begin{array}{c}\text{Effective cooperative area}\\\text{of the dual arm robot}\end{array} = \frac{\begin{array}{c}\text{Cooperative workspace of the dual arm robot}\\\text{with cooperative RI over 35}\end{array}}{\begin{array}{c}\text{Global workspace of}\\\text{a dual arm robot}\end{array}} \tag{2.5}$$

2.3.2 Trajectory based metrics

Workspace analysis has a limitation in that it cannot take into account the collision that occurred during the motion into the performance metrics. We propose new metrics using the trajectories collected from human motion. The metrics include how well the robot arm follows the human motion trajectories in terms of position error, success rate, and collision rate and how human-like the robot motion is. Trajectory generation is explained in section 2.3.2 and metric construction is illustrated in 2.3.2.

Trajectory collection from human motion capture data

TRINA robot system's scenarios include tasks conducted in hospitals and offices, i.e., pick up and place medical devices, open and close the drawer or cabinet, open the door, etc. We used TUM kitchen dataset [49] of human motion capture data tracked while the several table-setting tasks, i.e., opening and closing a door and a drawer, lowering an object, reaching towards an object, and taking an object. The kinds of everyday tasks interacting with the objects on the table or furniture are similar for most human environments, so we decided to use this motion data collected in the kitchen for our case.

Each frame of the dataset has been manually labeled to describe hand motion and indicate if the subject is moving(*'StandingStill'* or *'HumanWalkingMotion'*). We extract the trajectory performing the motion while standing still. We define the valid trajectories if the number of consecutive frames labeled as *'StandingStill'* exceeds 50. The reason that we only consider motion during standstill is that we solve IK for the joint angle of the robot arm, not including the degree of freedom of base movement and robot torso movement. We then transform the pose data to be located in front of the robot torso by subtracting the x-y pose of Pelvis joint, and adjusting the z-axis rotation. Finally, the trajectories are translated with x-offset(0.4m) due to the difference between human and robot torso.

Metric Design

For performance evaluation, we construct metrics that measure human-likeness, collision rate as well as tracking error while robot end-effector follows the trajectories.

Tracking trajectory A trajectory is represented in milestones recorded in each frame. Each milestone has information of orientation and position of finger extracted from human motion capture data. For each trajectory, we regard the milestones as target pose in order and use a local inverse kinematics solver to get the best-found configuration. For the first point, the local IK solver takes the behavior parameter as an initial guess. After that, the initial guess is the best-found pose generated after solving IK for the previous milestone. The local IK solver is computationally cheaper than the global IK solver, however, it is sensitive to initial guesses. Since bad initial guesses cause all points on the trajectory would be unreachable, we narrowed the sampling region from the configuration space to increase the success rate. In order to generate valid behavior parameters, we sample end-effector positions uniformly at random within a region in front of the chest, and orientation is set as same as the global frame. We then solve IK locally with the initial seed sampled uniformly at random in the configuration space. Then it ensures that the initial guess of robot arm configuration has no collision and its end-effector is inside the region of interest. We define the initial 6DOF configuration as a behavior parameter for each trajectory.

Collision The collision between two arms has not been investigated in reachability analysis. So we incorporate the collision rate into our metrics to guide the design optimization to a low collision rate. After solving IK at each milestone, we check the collision including self-collision, collision with objects, and terrain.

Human-likeness Due to the dissimilarity of structures between the robot and the human, human-likeness is hard to be measured. We choose to define a plane that contains finger, elbow, and shoulder joints for human and robot arms. Then the angle between normal vectors of planes is computed. The smaller difference in angle means that the robot arm's posture is human-like.

Formulation At each milestone, we constructed 4 metrics as below:

$$\text{Success Rate}(m_{1,i}^{L/R}) = \text{1 if IK solution exists and no collision. Otherwise 0.} \tag{2.6}$$

$$\text{Finger Position Error}(m_{2,i}^{L/R}) = \begin{array}{c}\text{Distance between the finger position and}\\ \text{the best found position of end-effector. (in m)}\end{array} \tag{2.7}$$

$$\text{Collision Rate}(m_{3,i}) = \text{1 if there's collision with Left arm or Right arm. Otherwise 0.} \tag{2.8}$$

$$\text{Structural Similarity}(m_{4,i}^{L/R}) = \begin{array}{c}\text{Angle between the normal vectors of the planes}\\ \text{defined by finger, elbow, and shoulder joints. (in rad)}\end{array} \tag{2.9}$$

where i is the index of milestone in a trajectory. We take average over all milestones in trajectory, and the length of trajectory is T. We transform equations to the objective function to be maximized. Then the metrics for each trajectory can be computed as shown in equations (2.10), (2.11), (2.12), and (2.13).

$$M_1 = \frac{1}{2T}\sum_{i=1}^{T} m_{1,i}^{L} + m_{1,i}^{R} \tag{2.10}$$

$$M_2 = \frac{1}{1 + \frac{1}{2T}\sum_{i=1}^{T}(m_{2,i}^{L} + m_{2,i}^{R})} \tag{2.11}$$

$$M_3 = 1 - \frac{1}{T}\sum_{i=1}^{T} m_{3,i} \tag{2.12}$$

$$M_4 = \frac{1}{1 + \frac{1}{2T}\sum_{i=1}^{T}(m_{4,i}^{L} + m_{4,i}^{R})} \tag{2.13}$$

2.3.3 Optimization method: Multi-Objective Bilevel Bayesian Optimization

The previous approach for the robot arm placement [10] coped with the multi-objective problem using the weighted sum method, and another one [17] narrowed down the angle range in a prioritized manner. Those approaches provide one solution but cannot investigate the whole range of the Pareto Front. Since multiple objectives generally have a conflicting relationship, it is hard to decide which objective is more important. In our previous work (Chapter 1), we proposed the co-design optimization method based on the multi-objective BO literature. Since the method can be applied to the multi-objective co-design problems containing environment-dependent behaviors, we decided to use this method to optimize the robot arm placement. Thus we can provide a wide range of choices to designers rather than giving a single design.

2.4 Experiment

2.4.1 Robot arm design problem

We designate 4 design parameters: distance between shoulder(range [0.1m 0.6m]), pan angle(range [0°, 90°]), tilt angle(range [0°, 180°]), and shoulder height(range [0.8m, 1.5m]). The shoulder distance and height have not been fully investigated as much as tilt or pan angle in the previous works for the robot arm placement. For the TRINA robot system, the distance between the shoulders and height of the shoulder should be considered since it affects the range of area where the robot occupies and reaches.

Baseline The common and cooperative reachability index may lead the optimization to generate designs with short shoulder distance since it brings larger region that both arms can reach. However, short shoulder distance may cause high collision rate during motion. To ensure there's no collision between the bases of each arm, we set the range of distance between shoulder to [0.2m, 0.6m]. Otherwise, the design parameters have same ranges with our method. In the optimization setting, three metrics shown in (2.2), (2.4), and (2.5) are dependent only on the design parameters so that the behavior parameters do not need to be chosen during the optimization. We generate 150 designs sampled uniformly at random in design space for the initialization and evaluate these designs. After initialization, we run 150 outer iterations with parameter setting $S = 20$. From two trials, 4 designs lie on the Pareto front in total, and Table 2.1 shows those designs. The designs have similar values of shoulder width and shoulder height. As shorter shoulder distance makes the common area larger, most of the designs on the Pareto front tend to have a shorter distance. Also, the pan angle of the designs has a narrow range: [85°, 90°].

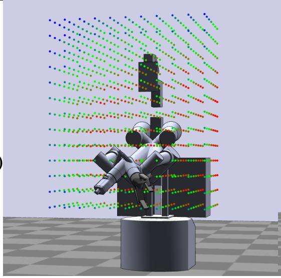
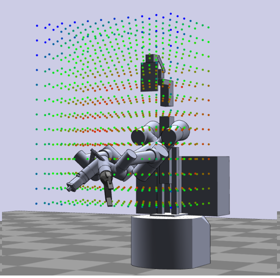
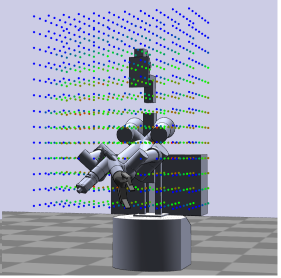
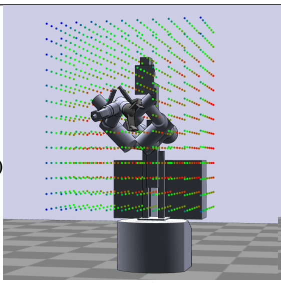
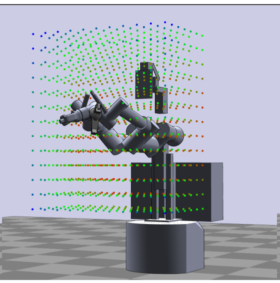
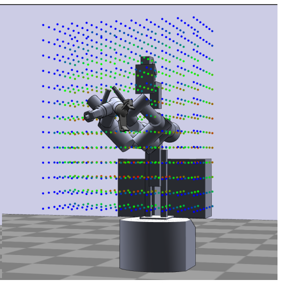
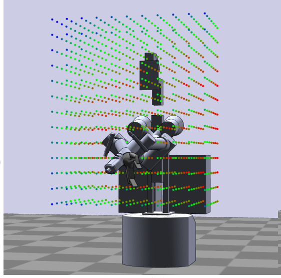
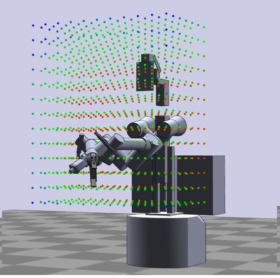
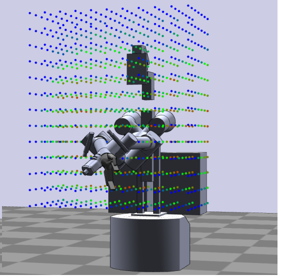
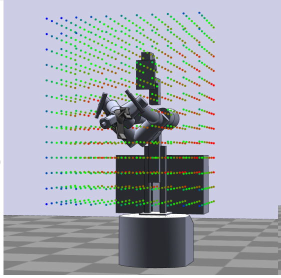
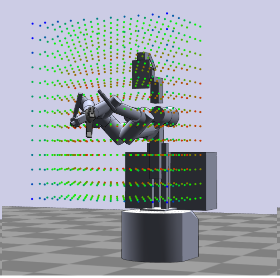
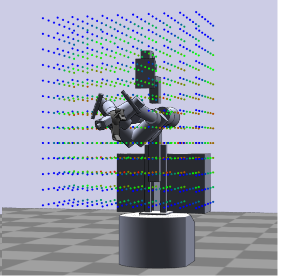
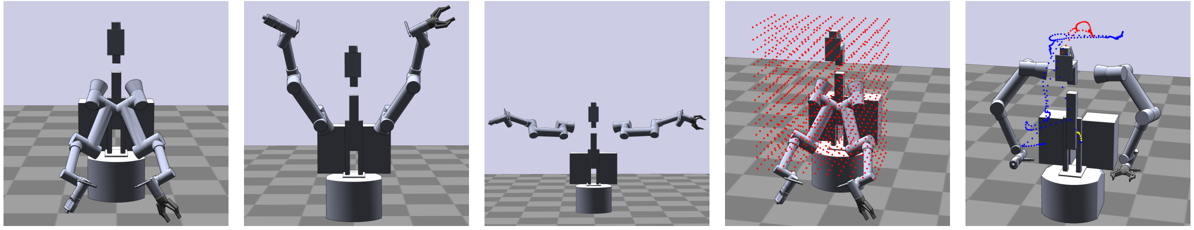
Metric Legend	High Dexterity(Left Arm) 0 🟦🟩🟥 1	Common area 0 🟦🟩🟥 1	Cooperative area 0 🟦🟩🟥 1
Design 1 (0.228m, 85.9°, 77.0°, 1.01m)			
Metric Value	**0.260**	0.579	0.444
Design 2 (0.220m, 87.0°, 99.3°, 1.02m)			
Metric Value	0.250	0.599	0.446
Design 3 (0.243m, 89.8°, 81.1°, 1.04m)			
Metric Value	0.256	0.595	0.443
Design 4 (0.227m, 90.0°, 92.2°, 1.09m)			
Metric Value	0.247	**0.632**	**0.451**

Table 2.1: Four robot arm designs generated by MO-BBO with Reachability-based metrics. Best performers highlighted in bold.

Our Method We constructed metrics as represented in (2.10), (2.11), (2.12), and (2.13). The metrics are evaluated in 11 trajectories from scene 13 of TUM Kitchen dataset. Then for each metric, we take the mean value over all trajectories as a score for each design and the dimension of objective space is 4. As the environment-dependent metrics are more than one, the inner optimization that selects design- and environment-specific decision becomes multi-objective problem.



(a) (0.3, 70, 50, 1.0) (b) (0.5, 45, 150, 0.8) (c) (0.6, 20, 100, 1.2) (d) Reachability (e) Trajectory

Figure 2.1: Robot arm placement problem. (a–c) show 3 illustrative designs with design parameters (shoulder width (m), pan (°), tilt (°), shoulder height (m)). (d) positions checked for reachability analysis. (e) trajectory from TUM Kitchen dataset

So we changed the inner optimization instead of using upper bound optimization used in [30]. We defined a new function that is almost same with the Likelihood of Pareto Expansion (LPE) metric scores used for the selection of the next design in [30].

Consider a candidate design $d \in D$. We want to select behavior from a set B of valid behavior samples. For each behavior $\theta \in B$, we evaluate the mean $\mu_{m,e}(d, \theta)$ and variance $\sigma^2_{m,e}(d, \theta)$ of $\mathbf{GP}_{m,e}$ at $[d, \theta]$. Given design d and environment e, the mean and covariance matrix of i^{th} metric is $[\mu_{1,e}, \cdots, \mu_{4,e}]^T$ and $\text{diag}(\sigma_{1,e}, \cdots, \sigma_{4,e})$. Then we draw K samples $\{\mathbf{m}^{(k)}(d, \theta) | k = 1, \ldots, K\}$ from the posterior Gaussian distribution. The following equation g is the ratio of the number of non-dominated samples by the current Pareto front $\mathbf{PF}$:

$$g(d, \theta) = \frac{1}{K} \sum_{k=1}^{K} \mathbb{1}[\mathbf{m}^{(k)}(d, \theta) \succ \mathbf{PF}] \tag{2.14}$$

The optimal behavior for design d is chosen by solving the following optimization:

$$\theta^* = \max_{\theta} g(d, \theta) \tag{2.15}$$

After selective θ^*, the next design candidate is selected in the same way with [30].

We generate 50 designs sampled uniformly at random in design space and for each design we evaluate metrics with 3 different valid behavior parameter samples for the initialization. Then we run 150 outer iterations with parameter setting $S = 20, |B| = 20$, where B is a set of valid behavior samples. 8 designs lie on the Pareto front. Among them, we select three designs with a success

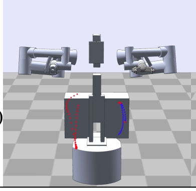
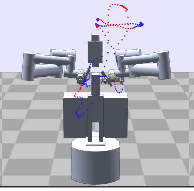
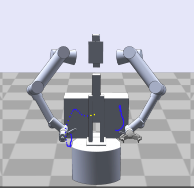
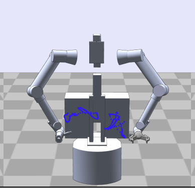
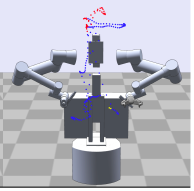
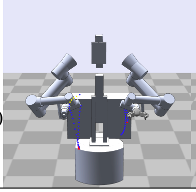
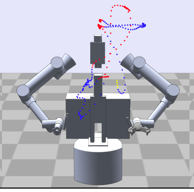
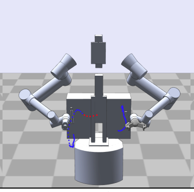
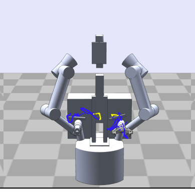
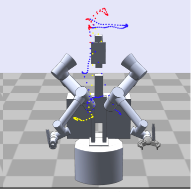
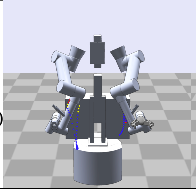
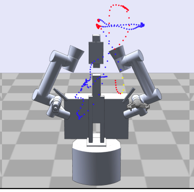
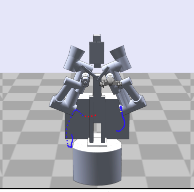
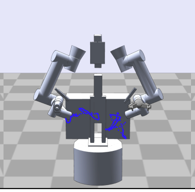
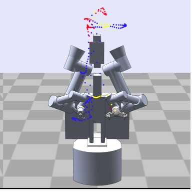
Trajectory	1	2	3	4	5
Legend	Blue: IK success & no collision / Yellow: IK success & collision / Red: IK failure				
Design 1 (0.441m, 15.4°, 98.3°, 1.29m)					
M_1 **(0.809)**	0.347	0.673	0.980	1.0	0.769
M_2 **(0.959)**	0.786	0.972	1.0	1.0	0.973
M_3 **(0.982)**	1.0	1.0	0.961	1.0	0.983
M_4 **(0.426)**	0.405	0.529	0.352	0.420	0.499
Design 2 (0.562m, 29.8°, 52.1°, 1.28m)					
M_1 **(0.851)**	0.754	0.647	0.951	0.906	0.605
M_2 **(0.989)**	0.982	0.947	0.993	1.0	0.972
M_3 **(0.911)**	0.932	0.971	1.0	0.902	0.714
M_4 **(0.548)**	0.567	0.567	0.492	0.487	0.565
Design 3 (0.385m, 25.6°, 26.7°, 1.39m)					
M_1 **(0.840)**	0.890	0.709	0.951	0.947	0.618
M_2 **(0.971)**	0.983	0.970	0.996	0.988	0.974
M_3 **(0.970)**	0.915	0.971	0.980	1.0	0.798
M_4 **(0.609)**	0.336	0.601	0.658	0.606	0.506

Table 2.2: Three robot arm designs generated by MO-BBO with Trajectory-based metrics. Five trajectories are selected among 11 trajectories used for the optimization. Bold indicates metric value averaged over 11 trajectories. Blue color indicates the structural similarity score.

rate averaged over 11 trajectories greater than 0.8, and Table 2.2 shows selected designs. Overall, three designs have high scores of M_1, M_2, M_3. However, the M_4 is significantly different for each design. The robot arm placement with a smaller tilt angle generates a more similar structure to the human arm.

2.4.2 Comparison

Test Environment For the comparison, we should investigate in new scenarios. Unseen trajectories are extracted from TUM Kitchen Dataset except for scene 13 that is already used for trajectory-based optimization. In total, 128 trajectories are used for the comparison.

Approach	Design Parameters	Success Rate	Finger Pose Error	Collision Rate	Structural Similarity
Reachability-Based	(0.218m, 85.9°, 77.0°, 1.01m)	0.275	0.947	0.649	0.438
	(0.217m, 90.0°, 92.2°, 1.09m)	0.345	0.964	0.646	0.386
Trajectory-Based	(0.441m, 15.4°, 98.3°, 1.29m)	0.751	0.959	0.929	0.423
	(0.562m, 29.8°, 52.1°, 1.28m)	0.673	0.931	0.880	0.530
	(0.385m, 25.6°, 26.7°, 1.39m)	0.665	0.952	0.804	0.597

Table 2.3: The test result of selected designs generated from two approaches: Reachability-based and Trajectory-based. The metric value is average over unseen 128 trajectories from 18 scenes of TUM dataset except for scene 13. For the designs generated by reachability based optimization, the metric values are averaged over three trials.

Selection of Initial configuration Initial configuration affects a lot when solving IK problem locally. We have to pick one behavior parameter carefully for the test trajectories. Since Reachability-based metrics are dependent only on design parameters, behavior parameters are not optimized. So we generate 100 valid samples and select one of them to be used for the test. Using each sample as an initial guess, we compute the four metrics defined in 2.3.2 for 11 trajectories that were already used for trajectory-based optimization. We select one with maximum hypervolume in the metric space, where hypervolume is computed as the multiplication of four metric values, as the reference point is origin $(0, 0, 0, 0)$. On the other hand, our method using trajectory-based metrics generates 11 behavior parameters according to 11 trajectories. Similarly, we compute hypervolume for those behavior parameters and pick one with the maximum hypervolume to be used as the initial guess for the test trajectories.

Result Design 1 and 4 in Table 2.1 and design 1, 2, and 3 in Table 2.2 are used for the comparison. The shoulder distance and height of optimized designs are larger in trajectory-based approach. Too short shoulder distance can be avoided in our method, since collision is checked over all milestones of dual-arm motions. It concludes that the collision is predicted more accurately in our method.

As shown in 2.3, the three designs optimized using trajectory-based metrics have lower values in M_1, M_2, M_3 for test trajectories than the scores in Table 2.2, whereas structural similarity metrics are consistent across different trajectories. Compared to those designs, the two designs optimized for the reachability-based metrics have a lower success rate. However, the finger pose error score is greater than 0.9, which means that in most milestones there exist IK solutions but collisions at the same time. The distance between two shoulders is too short in those cases, which leads to the high collision rate.

As we generate a wide range of solutions with multiple objectives, there is no ideal design. 3 selected designs generated in the trajectory-based approach have different characteristics. Design 1 in Table 2.2 is best for the success rate and collision rate, but the structural similarity score is the lowest. In contrast, design 3 has the highest structural similarity, while the success rate and collision rate are lower than design 1. Design 2 is in the middle of them. Eventually, the final choice among designs on Pareto front is up to the designers.

2.5 Conclusion

We presented new metrics for the robot arm placement problem. Our main idea was to use the human motion capture data to construct metrics that can evaluate the capability and the human-likeness of the dual-arm robot system. The trajectory-based metrics can effectively guide the optimization by avoiding designs that collide much and generating designs with a wide range of human-likeness.

In future work, the TRINA team plan to take into account the tilting motion of the torso and base movement when solving IK. If IK can be solved considering those DOFs, motion capture data with label *'HumanWalkingMotion'* can be used too. The failure case might be overcome and it may change the optimization result. Also, we selected the initial configuration for the test environments from a set of configuration candidates randomly sampled, which makes the performance worse than for trajectories used for the optimization. If we can parameterize environments, we might be able to select promising behavior parameters even for environments that are not used during the optimization.

In this work, we measure the human-likeness using the angle difference between the planes defined by the positions of joints to measure human-likeness. However, the human-likeness property is hard to formulate in the equation, since it depends on how humans recognize the robot's motion and posture. In future work, preference learning techniques such as [37], [8] can be applied to learn the human preference of the robot's posture and motion and use the learned function as a metric instead of the heuristically designed metrics.

Bibliography

[1] Albert Albers and Jens Ottnad. Integrated structural and controller optimization for lightweight robot design. In *2009 9th IEEE-RAS International Conference on Humanoid Robots*, pages 93–98, 2009.

[2] Ryo Ariizumi, Matthew Tesch, Kenta Kato, Howie Choset, and Fumitoshi Matsuno. Multiobjective optimization based on expensive robotic experiments under heteroscedastic noise. *IEEE Transactions on Robotics*, 33(2):468–483, 2017.

[3] Charles Audet, Jean Bigeon, Dominique Cartier, Sébastien Le Digabel, and Ludovic Salomon. Performance indicators in multiobjective optimization. *European Journal of Operational Research*, 292(2):397–422, 2021.

[4] Lorenzo Baccelliere, Navvab Kashiri, Luca Muratore, Arturo Laurenzi, Małgorzata Kamedula, Alessio Margan, Stefano Cordasco, Jörn Malzahn, and Nikos G. Tsagarakis. Development of a human size and strength compliant bi-manual platform for realistic heavy manipulation tasks. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5594–5601, 2017.

[5] M. Bagheri, A. Ajoudani, J. Lee, D. G. Caldwell, and N. G. Tsagarakis. Kinematic analysis and design considerations for optimal base frame arrangement of humanoid shoulders. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2710–2715, 2015.

[6] Syrine Belakaria, Aryan Deshwal, and Janardhan Rao Doppa. Uncertainty aware search framework for multi-objective bayesian optimization with constraints. *CoRR*, abs/2008.07029, 2020.

[7] Daniel M. Bodily, Thomas F. Allen, and Marc D. Killpack. Multi-objective design optimization of a soft, pneumatic robot. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1864–1871, 2017.

[8] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 89–96, New York, NY, USA, 2005. Association for Computing Machinery.

[9] Ian Char, Youngseog Chung, Willie Neiswanger, Kirthevasan Kandasamy, Andrew Oakleigh Nelson, Mark Boyer, Egemen Kolemen, and Jeff Schneider. Offline contextual bayesian optimization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[10] Chien-pin Chen, Pei-jui Wang, Wei-han Wang, Han-pei Wang, and Che-chien Chen. Developing industrial dual arm robot for flexible assembly through reachability map. *Technical Report*, 2015.

[11] Nick Cheney, Robert MacCurdy, Jeff Clune, and Hod 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*, pages 167–174, 2013.

[12] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models, 2018.

[13] Maíra Martins da Silva, Olivier Brüls, Wim Desmet, and Hendrik Van Brussel. Integrated structure and control design for mechatronic systems with configuration-dependent dynamics. *Mechatronics*, 19(6):1016–1025, 2009.

[14] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. In Marc Schoenauer, Kalyanmoy Deb, Günther Rudolph, Xin Yao, Evelyne Lutton, Juan Julian Merelo, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature PPSN VI*, pages 849–858, Berlin, Heidelberg, 2000. Springer Berlin Heidelberg.

[15] Stephane Doncieux, Nicolas Bredeche, Jean-Baptiste Mouret, and Agoston E. (Gusz) Eiben. Evolutionary robotics: What, why, and where to. *Frontiers in Robotics and AI*, 2:4, 2015.

[16] A. D. Dragan, K. C. T. Lee, and S. S. Srinivasa. Legibility and predictability of robot motion. In *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 301–308, 2013.

[17] J. Duan, Y. Gan, P. Cao, and X. Dai. The optimal solution for base frame installation of dual-arm robot. In *2019 IEEE 4th International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 546–552, 2019.

[18] Brian R. Duffy. Anthropomorphism and the social robot. *Robotics and Autonomous Systems*, 42(3):177–190, 2003. Socially Interactive Robots.

[19] Michael Emmerich. The computation of the expected improvement in dominated hypervolume of pareto front approximations. 01 2008.

[20] Michael T. M. Emmerich, André H. Deutz, and Jan Willem Klinkenberg. Hypervolume-based expected improvement: Monotonicity properties and exact computation. In *2011 IEEE Congress of Evolutionary Computation (CEC)*, pages 2147–2154, 2011.

[21] Erhan Erkut, Yilmaz Ülküsal, and Oktay Yeniçerioğlu. A comparison of p-dispersion heuristics. *Computers & Operations Research*, 21(10):1103 – 1113, 1994.

[22] Zhiwei Feng, Qingbin Zhang, Qiangang Tang, Tao Yang, and Jianquan Ge. Control-structure integrated multiobjective design for flexible spacecraft using moea/d. *Structural and Multidisciplinary Optimization*, 50(2):347–362, 2014.

[23] Michael J. Gielniak, C. Karen Liu, and Andrea L. Thomaz. Generating human-like motion for robots. *The International Journal of Robotics Research*, 32(11):1275–1301, 2021/04/27 2013.

[24] Michael J. Gielniak, C. Karen Liu, and Andrea L. Thomaz. Generating human-like motion for robots. *The International Journal of Robotics Research*, 32(11):1275–1301, 2013.

[25] David Ha. Reinforcement learning for improving agent design. *CoRR*, abs/1810.03779, 2018.

[26] 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*, 34(5):1240–1251, 2018.

[27] Daniel Hernández-Lobato, José Miguel Hernández-Lobato, Amar Shah, and Ryan P. Adams. Predictive entropy search for multi-objective bayesian optimization, 2016.

[28] Ian S Howard, James N Ingram, Konrad P Körding, and Daniel M Wolpert. Statistics of natural movements are reflected in motor errors. *Journal of neurophysiology*, 102(3):1902–1910, 09 2009.

[29] Junggon Kim, Kunihiro Iwamoto, James J. Kuffner, Yasuhiro Ota, and Nancy S. Pollard. Physically based grasp quality evaluation under pose uncertainty. *IEEE Transactions on Robotics*, 29(6):1424–1439, 2013.

[30] Y. Kim, Z. Pan, and K. Hauser. Mo-bbo: Multi-objective bilevel bayesian optimization for robot and behavior co-design. In *2021 IEEE International Conference on Robotics and Automation*, 2021.

[31] J. Knowles. Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, 10(1):50–66, 2006.

[32] Thomas Liao, Grant Wang, Brian Yang, Rene Lee, Kristofer S. J. Pister, Sergey Levine, and Roberto Calandra. Data-efficient learning of morphology and controller for a microrobot. *CoRR*, abs/1905.01334, 2019.

[33] M. Liarokapis, P. Artemiadis, Charalampos P. Bechlioulis, and K. Kyriakopoulos. Directions, methods and metrics for mapping human to robot motion with functional anthropomorphism. 2013.

[34] Minas Liarokapis, Charalampos P. Bechlioulis, Panagiotis K. Artemiadis, and Kostas J. Kyriakopoulos. Deriving Humanlike Arm Hand System Poses. *Journal of Mechanisms and Robotics*, 9(1), 01 2017. 011012.

[35] Abhijit Makhal and Alex K. Goins. Reuleaux: Robot base placement by reachability analysis. *CoRR*, abs/1710.01328, 2017.

[36] Ruben Martinez-Cantin. Funneled bayesian optimization for design, tuning and control of autonomous systems. *CoRR*, abs/1610.00366, 2016.

[37] Sahand Negahban, Sewoong Oh, Kiran K. Thekumparampil, and Jiaming Xu. Learning from comparisons and choices, 2018.

[38] Robert Pinsler, Peter Karkus, Andras Kupcsik, David Hsu, and Wee Sun Lee. Factored contextual policy search with bayesian optimization. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7242–7248, 2019.

[39] Oliver Porges, Roberto Lampariello, Jordi Artigas, Armin Wedler, Christoph Borst, and Maximo A. Roa. Reachability and dexterity: Analysis and applications for space robotics. In *Proceedings of the Workshop on Advanced Space Technologies for Robotics and Automation (ASTRA)*, 05 2015.

[40] C. Reyes and F. Gonzalez. Mechanical design optimization of a walking robot leg using genetic algorithm. In *Climbing and Walking Robots*, pages 275–284, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[41] E. B. Saff and A. B. J. Kuijlaars. Distributing many points on a sphere. *The Mathematical Intelligencer*, 19(1):5–11, 1997.

[42] R. Saravanan, S. Ramabalan, N. Ebenezer, and C. Dharmaraja. Evolutionary multi criteria design optimization of robot grippers. *Appl. Soft Comput.*, 9:159–172, 2009.

[43] Hiroyuki Sato, Kouhei Tomita, and Minami Miyakawa. Preferred region based evolutionary multi-objective optimization using parallel coordinates interface. In *2015 3rd International Symposium on Computational and Business Intelligence (ISCBI)*, pages 33–38, 2015.

[44] Charles B. Schaff, David Yunis, Ayan Chakrabarti, and Matthew R. Walter. Jointly learning to construct and control agents using deep reinforcement learning. *CoRR*, abs/1801.01432, 2018.

[45] Niranjan Srinivas, Andreas Krause, Sham M. Kakade, and Matthias W. Seeger. Gaussian process bandits without regret: An experimental design approach. *CoRR*, abs/0912.3995, 2009.

[46] Guoxin Sui and Yong Yu. Bayesian contextual bandits for hyper parameter optimization. *IEEE Access*, 8:42971–42979, 2020.

[47] Krister Svanberg. The method of moving asymptotes—a new method for structural optimization. *International Journal for Numerical Methods in Engineering*, 24(2):359–373, 1987.

[48] Kevin Swersky, Jasper Snoek, and Ryan P Adams. Multi-task bayesian optimization. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[49] Moritz Tenorth, Jan Bandouch, and Michael Beetz. The tum kitchen data set of everyday manipulation activities for motion tracking and action recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 1089–1096, 2009.

[50] Matthew Tesch, Jeff Schneider, and Howie Choset. Expensive multiobjective optimization for robotics. In *2013 IEEE International Conference on Robotics and Automation*, pages 973–980, 2013.

[51] N. Vahrenkamp, T. Asfour, and R. Dillmann. Robot placement based on reachability inversion. In *2013 IEEE International Conference on Robotics and Automation*, pages 1970–1975, 2013.

[52] Julian Whitman, Raunaq Bhirangi, Matthew J. Travers, and H. Choset. Modular robot design synthesis with deep reinforcement learning. In *AAAI*, 2020.

[53] 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*, pages 3229–3236, 2007.

[54] F. Zacharias, C. Schlette, F. Schmidt, C. Borst, J. Rossmann, and G. Hirzinger. Making planned paths look more human-like in humanoid robot manipulation planning. In *2011 IEEE International Conference on Robotics and Automation*, pages 1192–1198, 2011.

[55] Franziska Zacharias, Ian S. Howard, Thomas Hulin, and Gerd Hirzinger. Workspace comparisons of setup configurations for human-robot interaction. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3117–3122, 2010.

[56] Hongying Zhang, A. Senthil Kumar, Jerry Ying Hsi Fuh, and Michael Yu Wang. Design and development of a topology-optimized three-dimensional printed soft gripper. *Soft Robotics*, 5(5):650–661, 2018. PMID: 29985781.

[57] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, 2007.

[58] Allan Zhao, Jie Xu, Mina Konaković Luković, Josephine Hughes, Andrew Speilberg, Daniela Rus, and Wojciech Matusik. Robogrammar: Graph grammar for terrain-optimized robot design. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020.

[59] Jinhua Zheng, Guo Yu, Qiaofeng Zhu, Xiaodong Li, and Juan Zou. On decomposition methods in interactive user-preference based optimization. *Applied Soft Computing*, 52:952–973, 2017.