Crowdsourcing Interface Feature Design with Bayesian Optimization

John J. Dudley
Department of Engineering
University of Cambridge
Cambridge, United Kingdom
jjd50@cam.ac.uk

Jason T. Jacques
Department of Engineering
University of Cambridge
Cambridge, United Kingdom
jtj21@cam.ac.uk

Per Ola Kristensson
Department of Engineering
University of Cambridge
Cambridge, United Kingdom
pok21@cam.ac.uk

ABSTRACT

Designing novel interfaces is challenging. Designers typically rely on experience or subjective judgment in the absence of analytical or objective means for selecting interface parameters. We demonstrate Bayesian optimization as an efficient tool for objective interface feature refinement. Specifically, we show that crowdsourcing paired with Bayesian optimization can rapidly and effectively assist interface design across diverse deployment environments. Experiment 1 evaluates the approach on a familiar 2D interface design problem: a map search and review use case. Adding a degree of complexity, Experiment 2 extends Experiment 1 by switching the deployment environment to mobile-based virtual reality. The approach is then demonstrated as a case study for a fundamentally new and unfamiliar interaction design problem: web-based augmented reality. Finally, we show how the model generated as an outcome of the refinement process can be used for user simulation and queried to deliver various design insights.

CCS CONCEPTS

• **Human-centered computing → Systems and tools for interaction design**.

KEYWORDS

Interface Design; Optimization; Crowdsourcing

ACM Reference Format:
John J. Dudley, Jason T. Jacques, and Per Ola Kristensson. 2019. Crowdsourcing Interface Feature Design with Bayesian Optimization. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland UK*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3290605.3300482

CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

ACM ISBN 978-1-4503-5970-2/19/05…$15.00
https://doi.org/10.1145/3290605.3300482

1 INTRODUCTION

Without general guidance or analytical frameworks, user evaluation is critical to informing interface design. Performing this evaluation efficiently and identifying an optimum configuration is a fundamental goal of HCI. However, the process of optimizing the user interface is a non-trivial exercise given the typically noisy behavior of users and variability between users. Allowing the user to play a role in the optimization process is an attractive solution, particularly in instances where each evaluation is inherently user driven and the application is susceptible to user variability.

This paper examines Bayesian optimization as a potential tool in performing interface optimization in large scale, noisy user environments. Specifically we evaluate Bayesian optimization as an approach for online refinement of interface features through crowdsourced user participation. We apply Bayesian optimization to refine the parameters that determine the visual features and interaction behavior of a typical interface.

We present two illustrative experiments to demonstrate the process and its flexibility: 1) design of a 2D map search and review interface (such as encountered on a hotel booking site); and 2) design of a novel VR based search interface for the same task. We use these tasks and interfaces as a simple and familiar example to demonstrate the approach. Both interfaces are parameterized according to five design dimensions. We recruit users through crowdsourcing to identify ideal values for these parameters. The Bayesian optimization approach informs the selection of new test parameter values based on prior user performance, measured as the time to complete a discrete map search task. The experiment is structured to incorporate prior user data in batches in order to more clearly demonstrate an improvement in interface performance over time. As a baseline comparison we compare the Bayesian optimization approach with designs uniformly sampled from the bounded design space. This approximates arbitrary parameter settings chosen by a naive designer.

In addition to the two experiments described, we apply the procedure to an even more challenging mobile based Augmented Reality (AR) case study. This provides a practical demonstration of the approach and highlights its flexibility.

The key contributions of this paper are:

- An evaluation of Bayesian optimization for interface design refinement in two challenging design spaces.
- A demonstration of the approach in a highly novel web-based AR design case study.
- Implementation guidance for crowdsourcing interface design refinement using Bayesian optimization.

2 RELATED WORK

The broader challenge of actively supporting interface design refinement has been approached from a variety of perspectives. These efforts largely fall into three categories: model-based optimization, post hoc refinement, and online refinement. Model-based optimization methods support the designer at design time based on predictive models of the user [4, 20, 24, 27]. Keyboard layout optimization is a popular application of this approach. Applications such as MenuOptimizer [1], DesignScape [19] and Sketchplore [27] demonstrate how these approaches can also be explicitly embedded into design support tools.

Post hoc refinement is an offline strategy in which collected data is either used directly or fed to user models to refine the interface. Clearly this encompasses the much broader workflow of making design changes based on feedback and controlled experiments [9]. More relevant to the context of this paper, however, are efforts that formalize this approach [13, 23, 28]. Salem [23] demonstrates a structured approach to comparing and refining web landing page design alternatives using genetic programming while Liu et al. [13] explore optimal representations for mathematics pedagogy.

Online interface refinement, the category in which this paper falls, describes methods which actively change the interface based on some objective during or between interactions. This approach is readily applied in games where an optimal performance or engagement level might be achieved through game feature refinement [7, 14, 15]. Similarly, BIGnav [12] probabilistically fused inputs and prior information about locations on a map to improve navigation performance. Online refinement has also been explored in psychology (e.g. [17]) to obtain maximally informative experiments.

Bayesian optimization has significant potential in supporting this third strategy of interface refinement. Bayesian optimization is a machine learning technique that facilitates efficient exploration of complex or unobservable cost functions. The approach is particularly useful when evaluation of the cost function is expensive: e.g. slow computational models, or evaluations that involve a physical process. A detailed review of the approach and practical applications of Bayesian optimization is provided in [25]. Bayesian optimization has been applied in user interfaces for a range of applications. Brochu et al. [2] applied the technique within a preference gallery to allow users to evaluate alternate settings for rendering smoke. Other applications have involved maximizing user engagement in games [7], and optimizing individual user settings for a hearing device [18]. Snoek's doctoral thesis [26] provides a comprehensive investigation of Bayesian optimization for assistive technologies.

The incorporation of a Bayesian optimization approach into interface design by exploiting user interaction data is challenging due to typically high noise levels. Running large numbers of users through an interface that may be poorly designed in its first iteration can also be difficult for many designers. Fortunately, crowdsourcing has emerged as a viable source of large volumes of users willing to undertake interface testing in return for compensation. Crowdsourcing can offer large quantities of data at low cost [8]. Comparative studies, such as those by Heer and Bostock [6], have demonstrated crowdsourcing as a fast and effective method for gathering graphical perception data and provide results consistent with in-lab studies. There is also good precedence in interface research carried out using crowdsourcing [28]. Further, work by Komarov et al. [10] has shown that performance evaluation of user interfaces, carried out using both crowdworkers and in-lab participants, yields equivalent relative differences between experimental conditions.

Crowdsourcing has been employed at the intersection of interface design and Bayesian optimization to efficiently collect large numbers of user interactions. Koyama et al. [11] demonstrate the potential of Bayesian optimization to assist with visual feature optimization. They decompose the higher-order optimization problem into one-dimensional line searches that can then be allocated to crowdworkers: crowdworkers select the point on the slider that yields the best visual appearance. Koyama et al. [11] apply various quality control strategies to address aspects of subjectivity in this assessment. In this paper, we avoid subjectivity in crowdworker input by directly measuring task completion time: a summative reflection of the perceptual and interactive qualities of the interface. Khajah et al. [7] recruited crowdworkers in their evaluation of Bayesian optimization to find game parameters that maximize user engagement. As an indicator of engagement, they exploit crowdworker estimates of how much additional (unpaid) time others might spend playing the game. In this paper, we focus on utility and efficiency of the interface and target the most closely related performance metric for this purpose: task completion time.

The unique contribution of this paper is that we apply a Bayesian optimization approach to deliver refinement of a diverse set of design features directly based on actual user performance. Through demonstration in three different deployment settings, we also highlight the potential that this approach has as a design tool with good objectivity, versatility and comparatively low overhead.

3 APPROACH

The objective of this paper is to provide an accessible introduction to, and demonstration of Bayesian optimization for interface design refinement. Section 4 describes the technique of Bayesian optimization contextualized by the interface refinement problem. In contrast to prior work, we apply a formulation that is readily understood and applied: directly modeling the relationship between interface feature parameters and task completion time. In two illustrative design problems, we demonstrate that the technique actively refines the interface in very few execution cycles. A further case study serves to show how Bayesian optimization can be efficiently applied in totally unfamiliar applications and settings. Finally, we discuss the broader implications of our findings as they relate to the application of Bayesian optimization and highlight ancillary benefits of the approach.

4 BAYESIAN OPTIMIZATION

In this section we provide a brief overview of the basic principles of Bayesian optimization. For a detailed explanation and formulation see [26]. At the expense of completeness, we have endeavoured to provide a simple to understand explanation contextualized by the interface design problem.

Bayesian optimization works by exploiting a probabilistic model that has been fitted to describe some unknown function. In this study, for example, we seek to model how users perform when certain interface design features are varied. This function is 'unknown' as we have no way to reliably predict how user performance will be affected by changes to the interface.[1] The conventional approach in Bayesian optimization is to model the unknown function as a Gaussian process (GP). A GP is a distribution over functions and is specified by its mean function, $m(\mathbf{x})$ and covariance function, $k(\mathbf{x}, \mathbf{x}')$. These are essentially the function parallels of the mean and variance of a random variable. The function, $f(\mathbf{x})$, however, specifies the random variable at location, $\mathbf{x}$.

In the interface refinement task, we fit the GP using data obtained through observations of user performance. Throughout this paper we use task completion time as the observation value. An observation instance, representing a particular design configuration of the interface, has parameter values defined by $\mathbf{x}_i$. The crux of Bayesian optimization is to leverage the GP, fitted to a sequence of observations, $\{\mathbf{x}_{1:t}, \mathbf{f}_{1:t}\}$, to probabilistically determine what new point, $\mathbf{x}_{t+1}$, should be evaluated next. As part of fitting observation data to the GP, there are a number of subtle assumptions that must be made about the target function. One of these relates to how closely

[1]This is not to say that we cannot predict or estimate certain aspects of user performance. Techniques such as KLM and Fitts' Law might allow one to estimate the effect of changes in element sizes or placement. Such techniques, however, struggle when applied to simultaneous variation of multiple interface design parameters with nuanced factor interactions.

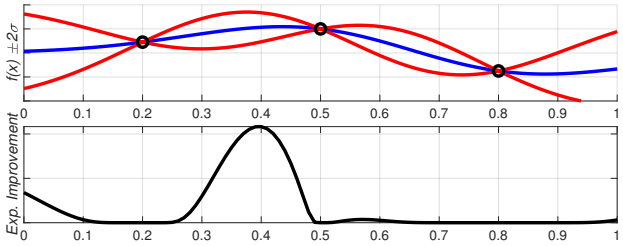

Figure 1: Illustration of Bayesian optimization in 1D. The top plot shows the Gaussian process approximation of the latent function over the design space. Three observations are shown (black circles) and the uncertainty around these points is visibly reduced. Below is the Expected Improvement over the design space: the potential that a new observation point has to improve upon the current best observation.

nearby points in the space are correlated. The covariance between two points is typically referred to as a kernel. In this study we employ the automatic relevance determination (ARD) kernel (see [21, pp. 106]). The kernel itself has parameters, typically referred to as hyperparameters, which can be thought of as describing the general shape of the function space independent of the data points. There are many possible choices of kernel but we select the ARD kernal for its simplicity and to demonstrate that the approach can perform effectively even in a rudimentary configuration.

Once the GP is fitted to the observation points, the next step is to determine which new point to sample based on some probabilistic guidance. This guidance comes from an acquisition function: a function that reflects the benefit of sampling a given set of parameter values. The literature provides many choices for acquisition function. We use a standard approach based on expected improvement (EI). The EI acquisition function can be thought of as the potential gain that can be obtained, relative to the current best observation, at a given new observation point. This assessment is based on the current model's mean and variance at that point. The concept of the acquisition function is illustrated in a one dimensional example in Figure 1.

Hyperparameters

As described above, Bayesian optimization is not completely free from the parameter selection problem. There are several hyperparameters in the ARD kernel which dictate the high level function shape. These are: $\theta = (\sigma_f^2, \sigma_n^2, l_1, ..., l_D)$ where D is the dimensionality. The signal variance, σ_f^2, is the variance in the signal without noise, i.e. the degree to which the signal varies over the space as a function of the inputs. If there are no observation points in a portion of the space, the standard deviation of the process is σ_f. The noise variance, σ_n^2, reflects the characteristics of the noise added to the underlying signal. As described by Rasmussen and

Williams [21, pp. 106], "the $l_1, ..., l_D$ hyperparameters play the role of characteristic length-scales; loosely speaking, how far do you need to move (along a particular axis) in input space for the function values to become uncorrelated."

Conveniently, there are effective methods for determining appropriate hyperparameter values. One of the main attractions of Gaussian processes for regression models is that the integrals are analytically tractable [21]. As such, it is possible to derive the expression for the marginal likelihood, i.e. the likelihood of the observations given the hyperparamters marginalized over the possible functions. Suitable hyperparameters are found by optimizing the marginal likelihood.

Implementation Specific Details

This section documents several details specific to the implementation of Bayesian optimization used in this study. As suggested in [21, p. 23], we rescale the observation values to have zero mean and unit variance. To do this in the absence of prior data, we require a coarse approximation of the distribution of typical observation values. In this paper, the observation values are task completion times. Completion time is approximately the product of the number of inspections and time per inspection. Such products approach a log-normal distribution. Based on initial pilot testing and for consistency across the experiments and case study, we assume a mean completion time of approximately 30 s. To normalize for unit variance we expect typical task times might vary between 15 s (half) and 60 s (double) which corresponds very approximately with a log-normal standard deviation of 0.7. These values could be refined through further pilot testing or using prior task data. Our results suggest, however, that coarse approximations yield adequate performance.

A further simplification for implementation purposes is the conversion of the continuous design space into a discrete one. This helps avoid the requirement to exhaustively search the space when optimizing the acquisition function. The approach involves evaluating a candidate list of sample points that provide representative coverage of the design space. Appropriate bounds for each parameter are chosen and this sets the limits of the hypercube. The candidate list is then constructed by sampling from the parameter hypercube using a Sobol sequence as described by [26]. We use 1000 candidates sampled in this way. While more candidates provides greater search resolution (at the cost of speed), we considered this value sufficient to demonstrate the approach. Unlike many optimization problems, we do not expect that a certain set of parameters will provide universally optimal performance. Far more likely in the case of varied human participants working on different platforms is that we see an ideal parameter region rather than a distinct peak. Therefore, fine-grained optimization of the parameters is not necessary. At this point it is also important to highlight a subtle distinction between advantageous exploration and convergence towards a singular 'optimal' design. This paper uses an optimization technique but is distinct from pure optimization. Rather, the expected behavior under the EI acquisition function is an emerging preference for selection from within a region of good designs (advantageous exploration). At some point, however, this advantageous exploratory behavior may be overridden by a preference for unexplored regions of high uncertainty exhibiting some potential for improvement.

A further deviation from more typical applications of Bayesian optimization is our batching approach. We hypothesize that the Bayesian optimization approach supports the identification of suitable parameter ranges while also reducing imposition on users. To make performance improvement due to parameter refinement testable, we incorporate prior participant data in separate batches. In other words, we complete a batch of tasks with multiple users and then feed in this performance data to provide prior information in subsequent batches. Note that this approach is not the same as selecting a set of parameter values to explore (e.g. [5]) as each user is allocated sample points independently of other users within the same batch. Within each batch and for each user, however, the standard process of Bayesian optimization also incorporates the individual user's prior performance. The hyperparameters are held constant during a batch and then updated based on all data up to and including the most recent batch. In the first batch with no prior data, the hyperparameters were all set to a nominal value (unity). Again, pilot testing or pre-existing data could inform the selection of appropriate values but we demonstrate that the approach can proceed even when naively initialized.

Fixed Baseline

A fixed baseline was introduced to serve as a common point of reference across the experimental batches that make up Experiments 1 and 2. Over both experiments, the condition was alternated for each subsequent participant. In the baseline condition, design parameters were uniformly sampled from the design candidate list. For a given participant, this sampling was without replacement such that a participant would not experience the same design twice.

Recall that the candidate list is constructed after first setting sensible bounds on the design parameters. This choice of baseline can therefore be thought of as testing parameters supplied by naive designers without prior experience or the ability to learn from prior data. This baseline is clearly conservative as even the most naive designer may be unlikely to choose certain design combinations, even if the individual parameter values are sensible. Nevertheless, this baseline serves as a useful reference point and an important check on population sampling effects.

5 EXPERIMENT 1: HOTEL SEARCH TASK

Our intent in this study is to evaluate the optimization approach in the context of a real word interface design problem. As an exploratory venture into this space we sought a relatively simple task that had good external validity but could still be experimentally controlled. We chose a map search interface such as encountered on most online hotel booking sites. Specifically, we refer to an interface in which hotel location *pins* are visualized on a map and the user reviews additional details (shown in overlaid *tooltips*) about each hotel by moving the cursor over the map. The task thus requires the user to find a hotel that meets specified criteria.

Given that the actual application is secondary to the demonstration of our approach in this paper, it is convenient that the map search and review task is generally familiar to users. We assume that the basic interface and interaction learning effects are small and we can avoid extensive explanation of the task. This means that the variation in the parameters which define the interface design are more likely to be the dominant factor influencing completion time.

The design of a map search interface is a useful demonstration application as it encompasses multiple non-trivial design dimensions. Consider, for example, the timeout period on hiding a tooltip after leaving a pin with the cursor. Setting this value to be too short may prevent the user from making a comparative evaluation while conversely, setting it too long or infinite may cause unnecessary obfuscation of the interface. The timeout period may be chosen by the designer through some self-testing or an informal user study but there is limited objective basis for assuming that value is appropriate for the broader user population. Furthermore, it is easy to imagine non-trivial interactions between this timeout period and other design parameters such as the distance threshold on initially showing the tooltip.

While the interface design space is obviously theoretically infinite, for practical purposes it is necessary to finitely parameterize the design space. For the purpose of this experiment, we constrain the parameterization of the design space to five dimensions. Constraining to five dimensions allows us to demonstrate utility in a non-trivial parameter selection problem while also maintaining interpretability of the design implications. The five design dimensions chosen are summarized below:

- *Distance:* threshold distance on cursor-to-pin for raising *show tooltip* event.
- *Delay*: timeout before responding to *show tooltip* event.
- *Decay*: timeout before responding to *hide tooltip* event after cursor exits distance threshold.
- *Size*: size of the hotel tooltip.
- *Opacity*: transparency of the hotel tooltip.

Clearly there are many other possible interface design features that could have been chosen. To illustrate the advantages of the presented approach, we have selected features that exhibit inherent trade-offs and expose non-trivial interactions with other features. The map search interface as developed for this experiment is illustrated in Figure 2.

Each hotel tooltip lists the name, thumbnail image, price, star rating and bed count. These details were set arbitrarily although effort was made to ensure there was a correlation between star rating and price as well as bed count and price, as per standard hotel room pricing practices. There were always 20 hotels indicated on the map.

Finding Hotels

Participants were instructed to find the hotel on the map meeting specified criteria. For example, the search criteria might say, “Find a hotel that is 3 stars and has 3 beds”. The participant must then search the map and review hotel details until they find the matching hotel. The search criteria were chosen so that there was only one hotel that matched the specified criteria.

Upon finding the matching hotel, the participant must click the *select* button, located on the tooltip, then click the submit button below the map. If an incorrect hotel was selected, the participant is informed of their error and forced to continue their search. A timer recorded the duration of the search task, and the counting timer was displayed on the top left of the map (see Figure 2). To avoid circumstances in which the interface parameters are so poor that they prevent the participant from finding the hotel or the participant is otherwise unable to complete the task, the task instance is automatically advanced after 90 s.

After submitting the correct hotel, the participant is presented with a results page. This page lists their completion time and the number of erroneous submissions made. If this was the first search task, no other information was presented

Figure 2: Hotel search task interface. The search criteria are displayed at the top of the interface. The tooltip details for four of the hotels are shown to the bottom left.

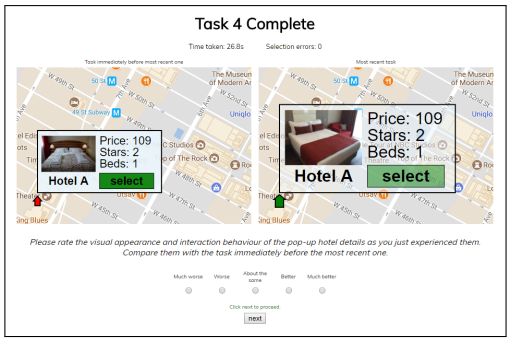

Figure 3: The comparative feature rating page. The left thumbnail shows the previous design configuration while the right shows the most recent configuration.

and the participant could just click the *next* button to move to the next iteration of the task. If this was the second or later task, the results page would also show two thumbnail maps with a single hotel (see Figure 3). These thumbnail maps presented the interface design as per the most recent parameter settings as well as the immediately previous parameter settings. The participant was then asked to rank their experience with the more recent parameter settings on a five point scale: much worse, worse, about the same, better, much better. After assigning their rating, the user could click the *next* button to move to the next task.

A total of 10 search tasks were presented to each participant, each with a different search criteria. The task order was randomly shuffled for each participant but the same 10 tasks were undertaken by all. Each task had a predefined hotel map layout. This layout was randomly generated originally to provide the distribution of hotels on the map but these layouts were then held constant for all users in the experiment described here.

Crowdsourcing Participants

This experiment was formulated as a Human-Intelligence Task (HIT) and participants were recruited through the Amazon Mechanical Turk service. No restrictions were placed on participant qualifications so any Mechanical Turk user, or *worker*, was able to complete the HIT. Workers were limited to completing the HIT only once so all participants in this experiment are unique.

Recall that the Bayesian optimization procedure demonstrated in this paper was applied in batches. Batch size was set to 20 participants. At 10 tasks per participant there were 200 unique parameter observations per batch. The procedure was executed for five batches in both the baseline and Bayesian optimization condition. Therefore, there were 200 unique participants in the experiment.

Participants were compensated US$1 for their participation. The HIT, including reading the introductory material

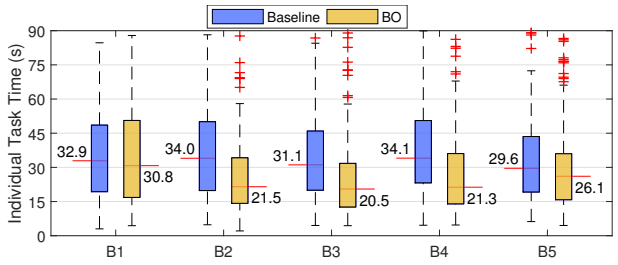

Figure 4: Boxplots of task completion time over the five batches for both conditions in Experiment 1. The red crosses indicate outliers based on $q_3 + 1.5 \times (q3 - q1)$.

and instructions, took approximately 10 minutes to complete. After completing all tasks, participants could elect to provide basic demographic information. In total, 79 specified female, 118 male, and three participants did not respond. Participant ages ranged from 21 to 65 with a median of 30.

6 EXPERIMENT 1: RESULTS

The batch results are summarized in Table 1 and Figure 4. Note that automatic advances of the task (i.e. where the task was not completed within 90 s) are excluded from these results although they are included as observations in the Bayesian optimization step. Note that discrete task time represents the time to complete a single search task and that each worker was presented with 10 search tasks. Each batch contained 20 workers so the n value reported in Table 1 indicates how may of the 200 tasks were actually completed.

We now review the results as obtained in chronological order of completion. In Batch 1, the boxplots of the baseline and Bayesian optimization conditions indicate very similar performance levels. A two-sample t-test on the log times reveals no significant difference between the samples (p=0.88).

This result is intuitive given that at this stage, the Bayesian optimization procedure has limited data upon which to model the parameter space. Given the parameter space is $\mathbb{R}^5$ there are insufficient samples to cover the corners of the hypercube. With insufficient data to make any firm assumptions about the space, the acquisition function typically encourages sampling that covers the space as broadly as possible.

Table 1: Median task times and completion counts in Exp. 1.

	Median Task Time (s) [n]	
Batch	*Baseline*	*BO*
B1	32.9 [165]	30.8 [166]
B2	34.0 [175]	21.5 [170]
B3	31.1 [161]	20.5 [187]
B4	34.1 [172]	21.3 [183]
B5	29.6 [165]	26.1 [191]

Batch 2 yields an improvement both relative to Batch 1 and its paired Baseline condition. The median completion time of 21.5 s represents a 30% reduction in median completion time compared to the same condition in Batch 1. A two-sample t-test reveals a significant difference between the Bayesian optimization and Baseline conditions ($p < 0.01$). This result suggests that the prior data incorporated from Batch 1 has encouraged the exploration of regions of the parameter space where there are actual performance improvements to be obtained. Batch 3 extends this further but with reduced gains. The median task time of 20.5 s is the lowest achieved and represents a 33% reduction relative to Batch 1.

It is interesting to note that Batches 4 and 5 remain significantly faster than the Baseline but are slightly elevated compared with the peak obtained in Batch 3. The interface improvements derived through the Bayesian optimization procedure are also evident in the increased task completion rates (n) (see Table 1), peaking at 95.5% in Batch 5 compared with 83% in Batch 1.

The performance plateau and subsequent increase in completion times can be explained by further exploration of the design space. As more observations are made in the region of good performance, this reduces the uncertainty in that region. The acquisition function used will always seek to maximize the expected improvement. At some point it is possible that although the predicted mean for a largely unvisited region is poor, the uncertainty in this region might promote its investigation. Although ultimately useful for modeling the complete design space, such explorations will manifest as poor batch aggregate performance. This problem is typically referred to as the exploration versus exploitation trade-off. At some point, it is better to 'exploit' the known region of good performance by taking finer and finer observations from within that region. As described in Section 4, a coarse candidate list was hypothesized to be appropriate for such interface refinement problems as fine parameter variation may not necessarily translate into noticeable difference in the interface. Nevertheless, there are alternative acquisition functions in the literature that better manage this transition between exploration and exploitation.

Interface Variation Ratings

An alternative perspective on the interface feature refinement procedure is provided by looking at the participant ratings made after each task (except for the first task where no relative comparison is possible). Figure 5 presents the rating proportions grouped based on three categories: Same (representing 'about the same' on the rating scale), Better (representing 'better' and 'much better') and Worse (representing 'worse' and 'much worse'). This reduction is done to improve the clarity of the observable trends.

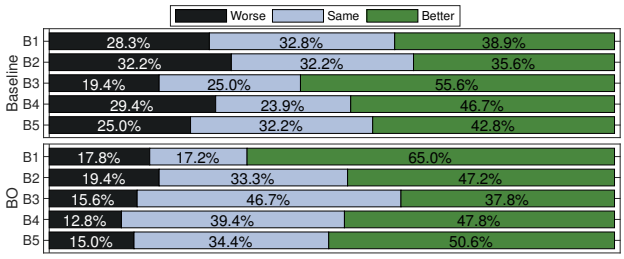

Figure 5: Rating proportions in Exp. 1.

In the Bayesian optimization condition, we expect that as the refinement process proceeds, the range of plausible parameter settings that offer potential improvements narrows. This trend is observable in Figure 5 where the Same counts steadily increase over batches 1-3. This largely comes at a cost of a smaller proportion of perceived improvements. The final two batches maintain a positive improvement bias but as observed in the median completion data, this does not translate into distinct performance improvement.

A further interesting result visible in Figure 5 is the high degree of variability for the Baseline condition. There is no reason that participants should perceive a task-to-task improvement in the interface in the Baseline condition yet there remains a positive bias over the batches. This observation may be due to a recency effect bias, but does suggest that when unconnected to a known model driving change such comparative preference data may be unreliable.

7 EXPERIMENT 2: MOBILE VR SEARCH TASK

The results from Experiment 1 suggest that Bayesian optimization presents a viable approach for refining 2D interface design parameters. As a subsequent test of the versatility of the procedure we investigate a less familiar and arguably more challenging interface design problem.

The 2D hotel search task was adapted to run as a mobile based quasi-virtual reality application. Rather than presenting the hotels on a 2D map surface, 3D hotel icons were displayed on an inclined map plane inside a rudimentary virtual environment. A screenshot of the task environment is presented in Figure 6. The view of the virtual environment is adjusted by using the mobile device as a window

Figure 6: Screenshot of the VR hotel search task in Exp. 2.

into the virtual world. It is important not to misconstrue the investigation of this particular interface as a suggestion for its practical use in a real-world hotel booking application. Rather, it serves as a demonstration of the Bayesian optimization approach in a more challenging interface deployment setting but with common design features.

For consistency, this task evaluated the same parameterization of the interface used in Experiment 1 with some minor adjustment for the differing interaction behavior: projected view-center-to-hotel distance threshold, show tooltip delay timeout, hide tooltip delay timeout, tooltip size and tooltip opacity.

As in Experiment 1, the participant must find the hotel that matches the specified criteria. Five batches were executed in both the Baseline and Bayesian optimization conditions. Batch size per condition was 20 participants as in Experiment 1. Participants could only complete the experiment once so all participants are unique. Note that participants who completed Experiment 1 were not prevented from completing Experiment 2. Of the 200 participants, 100 specified female, 97 male, one other, and two participants did not respond. Ages ranged from 18 to 64 with a median of 29. Participants were compensated US$1.20 for completing the HIT.

After each task, participants were again presented with the performance summary and rating page. Due to the confined screen space available in the mobile setting, no thumbnail reminders of the interface features were presented.

Performance Results

The results from Experiment 2 are summarized in Figure 7 and Table 2. The distribution of completion times in Batch 1 is broadly consistent between the Baseline and Bayesian optimization conditions. In Batch 2 there is an observable reduction in median completion time in the Bayesian optimization condition. There are further, but more marginal reductions in Batches 3 and 4. As in Experiment 1, there is a subsequent elevation in completion times in Batch 5. Again this is likely a consequence of disadvantageous exploration.

The difference in median completion time between Batch 1 and Batch 4 in the Bayesian optimization condition represents a reduction of 24.8%. The difference between conditions

Table 2: Median task times and completion counts in Exp. 2.

	Median Task Time (s) [n]	
Batch	*Baseline*	*BO*
B1	30.7 [170]	32.6 [164]
B2	34.3 [182]	25.5 [187]
B3	30.5 [182]	24.8 [192]
B4	36.3 [180]	24.5 [188]
B5	33.4 [181]	27.1 [186]

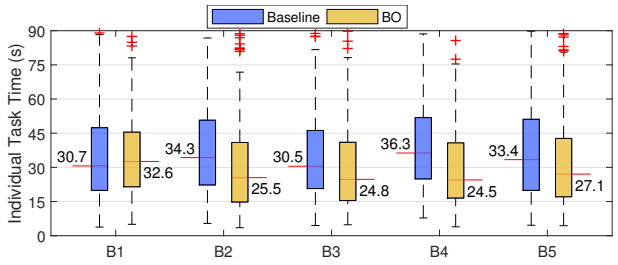

Figure 7: Boxplots of task completion time over the five batches for both conditions in Experiment 2.

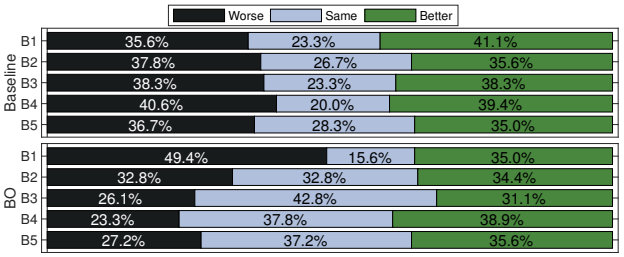

Figure 8: Rating proportions in Exp. 2.

in all but the first batch ($p = 0.31$) are significant ($p < 0.01$) based on two-sample t-tests applied to the log time.

Interface Variation Ratings

The post-task interface ratings are summarized in Figure 8, grouped into 'Worse', 'Same' and 'Better'. A distinct deviation from the results of Experiment 1 is the negative bias visible in the Bayesian optimization condition for Batch 1. Interestingly, this bias is reversed through batches 2 to 4.

Recall that, due to the limited screen space in the mobile setting, no thumbnail interface was presented to help participants recall the recent interface designs. It is reasonable to expect that this would make participant comparative judgments even more subjective and prone to error.

A consistent feature visible in both Figure 5 and Figure 8 is the peaking of 'Same' judgments in Batch 3. In both Experiments, Batch 3 is the batch where 'Same' ratings become the majority category. In both Experiments, Batch 3 is also the batch by which the most significant performance improvements have already been achieved.

8 DESIGN CASE STUDY: MOBILE AR TASK

Experiments 1 and 2 highlight the power of Bayesian optimization in delivering refinements to the interface in an objective and probabilistic fashion. As a further test we apply the refinement approach to a radically different and unfamiliar design problem. Furthermore, we remove the requirement to capture the Baseline condition and thus fully enable efficient parallelization of the technique through crowdsourcing. Experiments 1 and 2 serialized the participants in order to

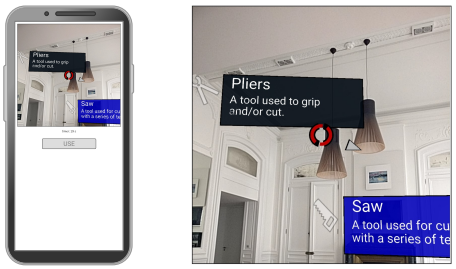

Figure 9: AR interface (left) and detail of scene (right).

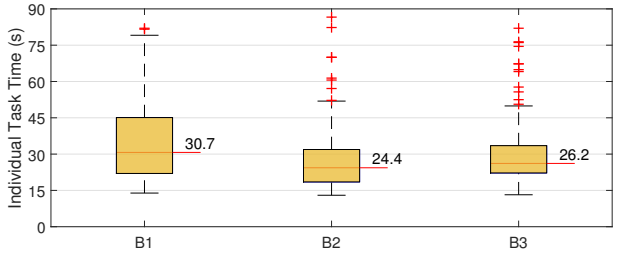

Figure 10: Boxplots of task completion time over the three batches run in the mobile AR task design case study.

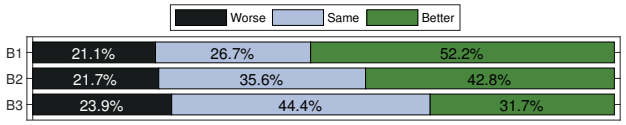

Figure 11: Comparative feature rating proportions in each batch for the AR case study.

ensure strictly alternating test conditions. Free from this constraint, it is theoretically feasible to launch a full batch for crowdworkers to complete in parallel.

The novel interface design challenge we tackle is the refinement of an interactive through-the-screen augmented reality (AR) experience. There is very limited research providing guidance on the design of interactive through-the-screen AR, particularly when deployed as a web application.

A design challenge particularly relevant to mobile device AR is gaze cueing. More specifically, exploiting the placement and behavior of virtual content to encourage users to look at certain target objects in the scene (whether physical or virtual). We developed a simple web application that constructed an AR experience in which users must review items in the scene then locate a specified item. For the purpose of the case study, this was framed as a task involving an inventory of virtual tools overlaid on the physical environment. To complete the task the participant must sequentially find and review all tools in the scene. After all tools were reviewed, an instruction would appear to find a specific tool. Figure 9 shows a screenshot of the tool finding AR interface.

The AR experience was constructed using the device camera feed as the background canvas for the virtual scene (built in A-Frame). To promote a more contextually connected experience, the coloration of each tool description panel would adapt to the physical background. As a rudimentary strategy, the description panel color was set based on the 180° offset from the mean hue of the background immediately behind the description panel. In addition, the text color would correspondingly switch between black or white depending on the perceived brightness of the description panel in order to promote readability [22].

This interface was parameterized into five design features. Some of these features are familiar with intuitive implications for task performance while others are highly novel with unpredictable influences. The bounds on parameters were set based on preliminary self-testing among the authors. Each of the design features is summarized below:

- *Background Timeout:* focus time required to mark tool as visited.
- *Foreground Timeout*: focus time required to return tool to foreground (required to select a specified tool).
- *Lightness Offset*: offset applied to panel color to discriminate between the in-focus and backgrounded state.
- *Gaze Guidance Grouping*: threshold on grouping the guidance arrows towards unvisited tools.
- *Opacity*: transparency of the description panel.

In Experiments 1 and 2 we observed limited subsequent benefit after two batches. The batch in which participant ratings of 'Same' became the dominant rating also appeared to provide a reliable indication of the point of limited subsequent improvement potential. Therefore, in this case study we use this indication as the trigger for terminating the refinement process.

Results

In total 60 participants completed the task (28 female, 30 male, 2 unspecified). Participant ages ranged from 19 to 46 with a median of 30.5. Participants were compensated US$1.20 for completing the HIT. The case study results are summarized in Figure 10. The median task completion times and completion counts [n] over Batches 1 to 3 were: 30.7 s [178], 24.4 s [182], and 26.2 s [156] respectively. Figure 11 plots the participant post-task rating proportions.

Based on the participant rating trigger proposed, the observation that the 'Same' category became the majority rating in Batch 3 suggested that the refinement procedure be concluded. Between Batch 1 and 2, the median completion time was reduced by 20.7%. There is then a marginal elevation in completion time between Batch 2 and 3. The plateauing of performance is reached earlier than in Experiments 1 and 2 but demonstrates that the proportion of 'Same' ratings provides an informative marker.

9 DISCUSSION

The results of this study highlight the value of Bayesian optimization as a method for supporting interface design through online user testing. We observed substantial reductions in task completion times in all three applications of the approach. The method is also able to accommodate the high levels of noise introduced by inter-user performance variability, inter-task variability and task learning effects. Under the rudimentary configuration of Bayesian optimization applied, the simple method for triggering termination based on perceived interface changes may be sufficient. More advanced configurations are available in the literature which help to better transition between exploration and exploitation.

A limitation of this work is that it is difficult to distinguish between truly optimizing design parameters and merely eliminating poorly performing regions of the design space. Pruning bad regions of the design space would yield the same result in terms of reduction in median task completion time. We plan to investigate whether this is the case but contend that bad-design rejection may in itself be useful.

The approach also has other practical advantages that may help streamline interface refinement exercises. Compared with alternative methods for evaluating the complete design space, Bayesian optimization can help to ensure that subsequent batches of the participant group benefit from the efforts of the previous group. Therefore, a well functioning optimization process of reasonable dimensionality should typically have worst case performance in the first batch. This may be useful in predicting task time or adjusting pay scales as the task becomes faster and easier to complete. It is likely that the variance in inter-user experience is also reduced.

There are several open questions we will address in future work. First, in this study we perform optimization based on performance metrics only. We hope to investigate how performance metrics might be complemented by pair-wise user ratings such as those collected. Second, this data driven approach might successfully integrate theory-driven design methods such as those described by [16]. In particular, such approaches might provide structured methods for selecting which parameters to refine and appropriate bounds. They may also be helpful in determining appropriate candidate resolution by reference to just noticeable differences. Third, our investigation is constrained to relatively low dimensionality so as to provide a simple demonstration of the method. The scalability of Bayesian optimization has received some attention in the machine learning community (see e.g. [3, 29]) but the implications of and procedures for dealing with a high dimensional design space remains as future work.

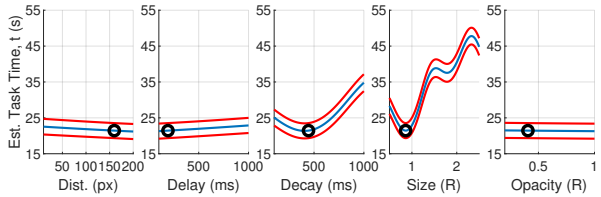

Figure 12: Sensitivity around optimal design candidate. Red lines show $\pm 2\sigma$. Note that this is the latent function model prediction which does not reflect the additive signal noise.

Querying the Design Model

An ancillary benefit of Bayesian optimization for interface feature refinement is that the procedure yields a model that has other potential uses. As an example, we can query the model to examine the sensitivity around the optimal design candidate identified in Experiment 1. The GP model incorporates all the collected samples and reflects the relationship between the design parameters and task completion time. It inherently accommodates and reflects the uncertainty in the sampling process. Figure 12 plots the variation in estimated task time as the parameter values are varied, one-at-a-time. From this plot we can observe that it may be possible to eek out further improvements by minor parameter tweaks. The plot also suggests that the *Decay* and *Size* parameters have the dominant effect on task time.

The generated performance model may also be used for simulation. The effect of proposed design changes may be estimated, not only to determine an approximate delta but also to estimate the anticipated distribution of performance. An extension of this idea is the ability to use the same model to identify parameters that minimize performance variation despite elevated average performance. This may indeed be a preferred outcome in some applications and use cases.

10 CONCLUSION

Bayesian optimization offers a powerful tool to support the objective refinement of interface designs. This has high potential value to designers given the low overhead of the approach and the fact that there is no subjective tuning required. The only real input required to initialize the process in the example presented is the setting of the bounds on the parameter values. We show that a batched approach to incorporating prior user performance data can deliver clearly detectable improvement in the interface with reductions in aggregate task completion time of between 33.3% and 20.7% in the deployments tested. Our results indicate that there is significant potential in this method as a generic means of supporting designers in objective and data driven refinement of their interfaces.

ACKNOWLEDGMENTS

This work was supported by EPSRC (grant EP/R004471/1) and the Trimble Fund. Supporting data for this paper is available at https://doi.org/10.17863/CAM.34781.

REFERENCES

[1] Gilles Bailly, Antti Oulasvirta, Timo Kötzing, and Sabrina Hoppe. 2013. MenuOptimizer: Interactive Optimization of Menu Systems. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 331–342. https://doi.org/10.1145/2501988.2502024

[2] Eric Brochu, Tyson Brochu, and Nando de Freitas. 2010. A Bayesian Interactive Optimization Approach to Procedural Animation Design. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '10)*. Eurographics Association, Goslar, Germany, 103–112. http://dl.acm.org/citation.cfm?id=1921427.1921443

[3] B. Choffin and N. Ueda. 2018. Scaling Bayesian Optimization up to Higher Dimensions: a Review and Comparison of Recent Algorithms. In *2018 IEEE 28th International Workshop on Machine Learning for Signal Processing (MLSP)*. 1–6. https://doi.org/10.1109/MLSP.2018.8517011

[4] Krzysztof Gajos and Daniel S. Weld. 2004. SUPPLE: Automatically Generating User Interfaces. In *Proceedings of the 9th International Conference on Intelligent User Interfaces (IUI '04)*. ACM, New York, NY, USA, 93–100. https://doi.org/10.1145/964442.964461

[5] Javier González, Zhenwen Dai, Philipp Hennig, and Neil Lawrence. 2016. Batch Bayesian Optimization via Local Penalization. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Arthur Gretton and Christian C. Robert (Eds.), Vol. 51. PMLR, Cadiz, Spain, 648–657. http://proceedings.mlr.press/v51/gonzalez16a.html

[6] Jeffrey Heer and Michael Bostock. 2010. Crowdsourcing Graphical Perception: Using Mechanical Turk to Assess Visualization Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 203–212. https://doi.org/10.1145/1753326.1753357

[7] Mohammad M. Khajah, Brett D. Roads, Robert V. Lindsey, Yun-En Liu, and Michael C. Mozer. 2016. Designing Engaging Games Using Bayesian Optimization. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 5571–5582. https://doi.org/10.1145/2858036.2858253

[8] Aniket Kittur, Ed H. Chi, and Bongwon Suh. 2008. Crowdsourcing User Studies with Mechanical Turk. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 453–456. https://doi.org/10.1145/1357054.1357127

[9] Ron Kohavi, Randal M. Henne, and Dan Sommerfield. 2007. Practical Guide to Controlled Experiments on the Web: Listen to Your Customers Not to the Hippo. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*. ACM, New York, NY, USA, 959–967. https://doi.org/10.1145/1281192.1281295

[10] Steven Komarov, Katharina Reinecke, and Krzysztof Z. Gajos. 2013. Crowdsourcing Performance Evaluations of User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 207–216. https://doi.org/10.1145/2470654.2470684

[11] Yuki Koyama, Issei Sato, Daisuke Sakamoto, and Takeo Igarashi. 2017. Sequential Line Search for Efficient Visual Design Optimization by Crowds. *ACM Trans. Graph.* 36, 4, Article 48 (July 2017), 11 pages. https://doi.org/10.1145/3072959.3073598

[12] Wanyu Liu, Rafael Lucas D'Oliveira, Michel Beaudouin-Lafon, and Olivier Rioul. 2017. BIGnav: Bayesian Information Gain for Guiding Multiscale Navigation. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, New York, NY, USA, 5869–5880. https://doi.org/10.1145/3025453.3025524

[13] Yun-En Liu, Travis Mandel, Emma Brunskill, and Zoran Popović. 2014. Towards Automatic Experimentation of Educational Knowledge. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 3349–3358. https://doi.org/10.1145/2556288.2557392

[14] J. Derek Lomas, Jodi Forlizzi, Nikhil Poonwala, Nirmal Patel, Sharan Shodhan, Kishan Patel, Ken Koedinger, and Emma Brunskill. 2016. Interface Design Optimization As a Multi-Armed Bandit Problem. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16)*. ACM, New York, NY, USA, 4142–4153. https://doi.org/10.1145/2858036.2858425

[15] MM Hassan Mahmud, Benjamin Rosman, Subramanian Ramamoorthy, and Pushmeet Kohli. 2014. Adapting Interaction Environments to Diverse Users through Online Action Set Selection. In *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*. 1–7. https://www.aaai.org/ocs/index.php/WS/AAAIW14/paper/view/8865

[16] Luana Micallef, Gregorio Palmas, Antti Oulasvirta, and Tino Weinkauf. 2017. Towards Perceptual Optimization of the Visual Design of Scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 23, 6 (June 2017), 1588–1599. https://doi.org/10.1109/TVCG.2017.2674978

[17] Jay I. Myung, Daniel R. Cavagnaro, and Mark A. Pitt. 2013. A Tutorial on Adaptive Design Optimization. *Journal of Mathematical Psychology* 57, 3 (2013), 53 – 67. https://doi.org/10.1016/j.jmp.2013.05.005

[18] J. B. B. Nielsen, J. Nielsen, and J. Larsen. 2015. Perception-Based Personalization of Hearing Aids Using Gaussian Processes and Active Learning. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 23, 1 (Jan 2015), 162–173. https://doi.org/10.1109/TASLP.2014.2377581

[19] Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. 2015. DesignScape: Design with Interactive Layout Suggestions. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 1221–1224. https://doi.org/10.1145/2702123.2702149

[20] Seonwook Park, Christoph Gebhardt, Roman Rädle, Anna Maria Feit, Hana Vrzakova, Niraj Ramesh Dayama, Hui-Shyong Yeo, Clemens N. Klokmose, Aaron Quigley, Antti Oulasvirta, and Otmar Hilliges. 2018. AdaM: Adapting Multi-User Interfaces for Collaborative Environments in Real-Time. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 184, 14 pages. https://doi.org/10.1145/3173574.3173758

[21] Carl Edward Rasmussen and Christopher K. I. Williams. 2006. *Gaussian Processes for Machine Learning*. MIT Press.

[22] C. Ridpath and W. Chisholm. 2000. Techniques For Accessibility Evaluation And Repair Tools. (2000). https://www.w3.org/TR/AERT

[23] Paulo Salem. 2017. User Interface Optimization Using Genetic Programming with an Application to Landing Pages. *Proc. ACM Hum.-Comput. Interact.* 1, EICS, Article 13 (June 2017), 17 pages. https://doi.org/10.1145/3099583

[24] Sayan Sarcar, Jussi Joklnen, Antti Oulasvirta, Chaklam Silpasuwanchai, Zhenxin Wang, and Xiangshi Ren. 2016. Towards Ability-Based Optimization for Aging Users. In *Proceedings of the International Symposium on Interactive Technology and Ageing Populations (ITAP '16)*. ACM, New York, NY, USA, 77–86. https://doi.org/10.1145/2996267.2996275

[25] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas. 2016. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* 104, 1 (Jan 2016), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

[26] Jasper Snoek. 2013. *Bayesian Optimization and Semiparametric Models with Applications to Assistive Technology*. Ph.D. Dissertation. University of Toronto, Toronto, Canada.

[27] Kashyap Todi, Daryl Weir, and Antti Oulasvirta. 2016. Sketchplore: Sketch and Explore with a Layout Optimiser. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (DIS '16)*. ACM, New York, NY, USA, 543–555. https://doi.org/10.1145/2901790.2901817

[28] Michael Toomim, Travis Kriplean, Claus Pörtner, and James Landay. 2011. Utility of Human-computer Interactions: Toward a Science of Preference Measurement. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2275–2284. https://doi.org/10.1145/1978942.1979277

[29] Ziyu Wang, Frank Hutter, Masrour Zoghi, David Matheson, and Nando De Freitas. 2016. Bayesian Optimization in a Billion Dimensions via Random Embeddings. *Journal of Artificial Intelligence Research* 55 (2016), 361–387. https://doi.org/10.1613/jair.4806