Personalization of a Mid-Air Gesture Keyboard using Multi-Objective Bayesian Optimization

Junxiao Shen* Jinghui Hu† John J. Dudley‡ Per Ola Kristensson§

University of Cambridge

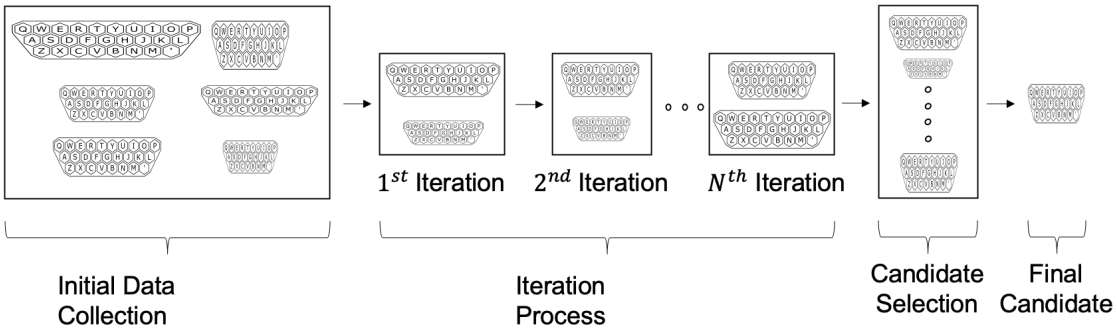


Figure 1: The personalization process of the mid-air gesture keyboard using multi-objective Bayesian optimization (MOBO) on layout size. Six different layout candidates with varying sizes are generated and evaluated by the user, then the process will introduce another two candidates for evaluation and this forms one iteration. Such an iteration needs to be repeated four to five times. Finally, a selection of candidates exhibiting optimal speed-accuracy trade-offs are generated. The user can then select the final optimal candidate based on their own preferences.

Abstract

We present AdaptiKeyboard, a mid-air gesture keyboard that uses multi-objective Bayesian optimization to adaptively change layout size to simultaneously optimize speed and accuracy. Gesture keyboards are well suited for enabling mid-air text entry in augmented reality (AR) due to their relative robustness to articulation inaccuracy. However, transplanting gesture keyboards to AR involves a larger design and operational space compared to touchscreen interactions. One potential advantage of this larger design and operational space is that mid-air keyboards presented in AR can be more versatile than their touchscreen equivalents. A key component of a mid-air gesture keyboard is the layout size, which can be made adaptive in order to optimize text entry speed and accuracy at the individual user level. This adaptive personalization can refine the keyboard design to reflect the differences users exhibit in motor behaviors and personal preferences. In this paper, we propose a multi-objective Bayesian optimization approach for adapting the layout size of a mid-air gesture keyboard to individual users. We show that this process can deliver a 14.4% improvement in speed and a 13.8% improvement in accuracy relative to a baseline design with a constant size derived from the default system keyboard on the HoloLens 2.

Index Terms: Human-centered computing—Human computer interaction (HCI)—Interaction techniques—Text input; Human-centered computing—Interaction design—Interaction design process and methods—User interface design

*e-mail: js2283@cam.ac.uk
†e-mail: jh2265@cam.ac.uk
‡e-mail: jjd50@cam.ac.uk
§e-mail: pok21@cam.ac.uk

1 Introduction

Augmented reality (AR) head-mounted displays (HMDs) provide users with experiences that seamlessly blend digital and physical content. However, AR increases the operational space available to users compared to conventional laptops and mobile devices, and has the capacity to dramatically increase the display space. The resulting increased design space leaves designers with considerable freedom to tune and optimize their designs for their AR applications. Meanwhile, 3D operational space also injects new interaction considerations compared to the more familiar 2D interaction surfaces. Further, different users exhibit different behaviors and preferences, and an increased operational space amplifies this consideration. As a consequence, an optimized design for one user may not be suitable for another. Therefore, application design in AR can likely benefit from a process that adapts the design in response to user behavior to optimize the performance for individual users. The process of adapting the design to an individual user can be framed as the *personalization* of the application.

A mid-air gesture keyboard [29, 32, 56] is a generally efficient text entry method [43] suitable for text entry in augmented reality (AR) and virtual reality (VR) [1, 18, 50, 54]. In particular, a gesture keyboard is an example of an intelligent text entry method [30] that satisfies two important criteria for mainstream success [31]: good performance and high similarity to existing methods users are already familiar with. The mid-air gesture keyboard, operating under an enlarged operational space and exploiting the available 3D operational space by allowing freehand 3D trajectory-based input, is especially suitable for personalization. However, adaptive design optimization of the mid-air gesture keyboard has not been previously studied.

The full design space of a mid-air gesture keyboard includes a multitude of continuous and categorical parameters. Keyboard size, inclination angle, and delimitation distance are some examples of continuous design parameters, while key shape and color are examples of discrete parameters. As such a keyboard offers a vast design space for personalization. Performing personalization across

all design parameters is impractical or perhaps even impossible. Therefore, two authors conducted a preliminary exploration of parameters with the aim to narrow down the design space. Through observation they uncovered prominent differences in performance when using different keyboard sizes, hence this paper focuses on the personalization of keyboard width and aspect ratio as design parameters.

The objectives of the optimization procedure are to maximize text entry speed and accuracy. However, fast gesture performance may sacrifice accuracy due to the inherent trade-off between speed and accuracy. Therefore, we formulate this personalization challenge as a multi-objective optimization problem since we are optimizing two objective functions simultaneously where the two objectives may conflict with each other. The goal of a multi-objective optimization problem is to identify the *Pareto front*—a set of optimum trade-offs, where an improvement in one objective means deteriorating another objective. Furthermore, this problem involves a function that is difficult and expensive to evaluate. This function is defined by the input being the different keyboard sizes and the output being the resulting speed and accuracy. We propose a multi-objective Bayesian optimization approach which is a data-efficient optimization method for global optimization of black-box functions to solve this multi-objective optimization problem.

In this paper, we present AdaptiKeyboard—a mid-air gesture keyboard that can adapt the width and aspect ratio of the keyboard to optimize speed and accuracy. The keyboard's geometry and interaction design are derived from the system gesture keyboard on the HoloLens 2. In a user study with 12 participants we demonstrate that AdaptiKeyboard can deliver a 14.4% improvement in speed and a 13.8% improvement in accuracy compared to a baseline keyboard that has a fixed size. The fixed-size keyboard is the same as the default size of the system keyboard on the HoloLens 2. From the results of the user study, we observe that different individuals have different optimal settings for keyboard width and aspect ratio. This observation emphasizes the importance of making a keyboard adaptive and thus further motivates this work.

In summary, this paper makes the following contributions:

1. We introduce AdaptiKeyboard: an adaptive keyboard that utilizes a multi-objective Bayesian optimization (MOBO) approach to adapt the width and aspect ratio for different users. To our knowledge, this is the first time that a mid-air gesture keyboard has been made adaptive by altering the layout size using MOBO.

2. We report the results of an empirical evaluation of the adaptive keyboard and demonstrate the improvement of this adaptive keyboard in both speed and accuracy compared to a baseline keyboard with constant size.

2 Related Work

In this section, we briefly review the related work to provide an understanding of adaptive interface design, multi-objective optimization and Bayesian optimization in design, and various optimization methods used in the design of text entry methods.

2.1 Adaptive Interface Design through Optimization

Adaptive interfaces [4, 21, 40, 46] aim to dynamically and autonomously enhance the user experience by altering interface characteristics to facilitate adaptivity in various dimensions such as the environment [17], user interests [33] or user capabilities [14], based on modeling the prior user interactions. Adaptive interface design using optimization, in contrast to design optimization which is a one-shot process, is an integrative approach that leverages data modeling to optimize on the fly. Many attempts have been made in the development of adaptivity strategies either based on expert knowledge [25] or generated from the collections of user data using machine learning [23, 47]. More recently, neural network-based methods for adaptive design optimization have been demonstrated [47]. Typically, this involves optimizing an objective function combining various layout attributes for better usability performance such as in search and pointing [48] and menu search [47]. Seonwook et al. [41] incorporated combinatorial optimization to adaptively allocate content and layouts for real-time collaboration across multiple devices. Lindlbauer et al. [35] presented an optimization-based approach for Mixed Reality (MR) systems to automatically control which applications are displayed, how much information they show and where they are placed in real-time. Evangelista Belo et al. [20] proposed a toolkit to visualize the interaction cost in a 3D user interface and implemented a prototype to demonstrate how creators can use this toolkit to design adaptive ergonomic user interfaces. However, there is no research to our knowledge that has studied the personalization of mid-air keyboards in augmented reality or virtual reality.

2.2 Multi-Objective Optimization in Design

Multi-objective optimization involves optimizing two or more objective functions simultaneously where the multiple objectives are often conflicting with each other, that is, optimizing one objective may deteriorate other objectives. In contrast to single-objective optimization where the superiority of a solution over other solutions is simply selected by ranking the objective function values, the solutions in a multi-objective optimization problem are determined from the Pareto front where there exist solutions in which none of the objectives can be improved without sacrificing at least one of the other objectives. These solutions are called Pareto optimal solutions, and these Pareto optimal solutions can be achieved through various optimization algorithms such as grid-based methods [19], evolutionary algorithms [12, 42] and Bayesian optimization [7]. Purvis et al. [42] treated the customization of document creation according to certain design criteria as a multi-objective optimization problem and used a genetic algorithm. Chan et al. [7] leveraged Bayesian optimization to investigate the positive and negative qualities of designer-led and optimization-driven design in a study that included a multi-objective optimization problem with novice designers. They found a superior outcome delivered from the optimization-driven design. In this paper, we use Bayesian optimization as the multi-objective optimization algorithm not only because it is a sample-efficient method for optimization, but also because it shows great potential in HCI design as detailed in Section 2.3

2.3 Bayesian Optimization in Design

The strength of Bayesian optimization (BO) is recognized as the ability to statistically model black-box objective functions and find the extrema in an efficient way. Greenhill et al. [24] explored the use of BO in an experimental design methodology, namely Adaptive Design Optimization (ADO). It was proposed to tackle and speed up experimental design optimization problems by using an intelligent data collection scheme. Adaptive user interface design could also benefit in a similar manner from using BO as it integrates user modeling and intelligent data collection. AdaptiFont [27] used BO in a similar way to our work. It progressively refined the text font for each individual using BO to find the optimized font space which contributed to optimal reading speed. Dudley et al. [16] leveraged crowdsourcing and Bayesian optimization to assist with interface design across diverse deployment environments. However, the aforementioned studies only have one objective whereas we optimize two potentially conflicting objectives at the same time, making the task particularly challenging.

2.4 Optimization in Text Entry

Various optimization methods have been applied to text entry techniques. These methods include gaze-assisted selection-based text en-

try [36], model-based strategies to simulate human data for interface development in evaluating touchscreen keyboard designs [26, 44] and alternative keyboard layouts [11,38,55]. The Qwerty layout has been adopted almost universally on modern devices despite other optimized keyboard layout alternatives such as the Metropolis [55], Opti [38] and Dvorak [11] layouts providing notionally faster entry rates. However, these layouts suffer the alienation problem that introduce issues with learnability, and thus prevent them-self from being adopted on modern devices. Similar work has also been done on touchscreen gesture keyboards. Smith et al. [45] focused on gesture clarity to reduce the ambiguity and similarity between identical or similar gestures by rearranging the keys to reduce error rates in gesture recognition. Although Smith et al. [45] considered the similarity of the optimized keyboard to the standard Qwerty keyboard using a multi-objective optimization method, none of these optimized keyboards have gained mainstream adoption due to issues with learnability. Bi et al. [5] suggested that alternation of only one pair of keys will significantly increase expert typing speed without a significant detrimental impact on learnability. Nevertheless, there are no notable examples of adoption. Gesture keyboard design optimization has been well explored in touchscreens but is still in its infancy in the AR/VR environment. The AR/VR design space introduces unique additional challenges given the extra dimensionality of interaction and the extensive display space available.

In this paper, we optimize the design of the mid-air gesture keyboard by altering the layout size. Despite the aforementioned research optimizing the keyboard by proposing new layout alternatives, the performance metrics they optimize still provide general guidance on what we should optimize during the personalization process. Both Smith et al. [45] and Bi et al. [5] used gesture clarity to quantify gesture typing accuracy and gesture speed to estimate how quickly users can gesture type on a keyboard layout. However, these metrics do not apply to mid-air gesture keyboards as the interaction paradigm may be fundamentally different due to the extra dimensionality. For example, the gesture speed metric is based on the CLC (curves, line segments, and corners) model by Cao et al. [6] which was designed to predict the amount of time it takes for a person to make an arbitrary pen stroke gesture, and this CLC model may not generalize to stroke gestures on a mid-air gesture keyboard. Moreover, text entry throughput, proposed by Zhang et al. [57], is a theoretically unified speed-accuracy metric built on Shannon information theory. However, both speed and accuracy provide essential practical insight and we optimize these two objectives to prevent any information loss during the optimization process.

3 Personalization as a Multi-Objective Optimization Problem

We formulate the problem of the adaptive keyboard as a multi-objective optimization problem to provide a better understanding of how we address the personalization of the mid-air gesture keyboard size.

We have a set of allowable design parameters, C, available to an adaptive strategy for modifying the design of the gesture keyboard. We choose the aspect ratio and width to be the design parameters as these two parameters can in general determine the overall size of the keyboard, which are critical factors affecting users' eventual speed and accuracy. We propose a grid of design candidates which forms the design space of Bayesian optimization. Here the design candidate is defined by the width and aspect ratio, and we will use the term *candidate* to denote design candidate for brevity. We use the height-to-width ratio to represent the aspect ratio. The minimum and maximum height-to-width ratios are 0.1875 and 0.5 respectively. The minimum and maximum acceptable keyboard widths are 100 mm and 600 mm respectively. These values are determined from preliminary experiments. We normalize the width (W) and height-to-width (HW) ratios by dividing by the maximum bound, such that the normalized parameter values have a maximum of 1. There are other parameters in the keyboard design that can be optimized in addition to the parameters considered in this paper. However, our preliminary experiments suggest that these parameters are not as critical as keyboard size.

In addition, we have a process performance measure, R, that specifies the performance of the keyboard with design parameters C under user inputs U. As we previously discussed in Section 2.4, we focus on speed and accuracy. In this paper, we measure speed (entry rate) in words per minute (WPM), which is the number of words entered per minute with each word standardized to be five characters long, including spaces. We quantify accuracy as one minus the Character Error Rate (CER) and then multiply it by 100 to arrive at a percentage value. CER is the minimum number of character insertion, deletion and substitution operations that transform the response text into the stimulus text, divided by the length of the stimulus text.

Our personalization task is thus a multi-objective optimization (MO) problem as the goal is to optimize the two potentially conflicting performance measures: speed and accuracy. Note that improving one objective may result in worsening another objective. We aim to find the Pareto optimal solutions which are discussed in Section 2.2.

First, to tackle this MO problem, we formalize this design problem as a function $f(\mathbf{C})$ where $\mathbf{C}$ is the feasible design space. It is inherently challenging to optimize the design of a keyboard because: 1) the design has two parameters and two objectives, and the exact mappings $f(\cdot)$ are unknown; and 2) each design takes time to evaluate with users. Furthermore, we can only observe function values $f(\mathbf{c})$ when making an evaluation at an input location $\mathbf{c}$ without knowing the first- or second-order derivatives at that location. This prevents the application of first- and second-order methods, such as gradient descent, Newton's method, or quasi-Newton methods. Problems with this property are referred to as "derivative-free" [22]. A data-efficient approach to solving such a black-box and derivative-free optimization problem, that is $\max_{\mathbf{c}\in\mathbf{C}} f(\mathbf{c})$, is to use Bayesian optimization, which can alleviate the aforementioned challenges. Bayesian optimization offers a more sample-efficient method for solving MO problems compared to some multi-objective optimizers, which require a large number of function evaluations [12].

4 The Bayesian Optimization Approach

Before we introduce the proposed multi-objective Bayesian optimization approach, we present a brief introduction on Bayesian optimization.

Bayesian optimization is a class of machine-learning-based optimization methods focused on solving $\max_{\mathbf{c}\in\mathbf{C}} f(\mathbf{c})$, where $\mathbf{C}$ is the feasible design space. Bayesian optimization consists of two components: a surrogate model, which is an approximation of the objective function $f(\mathbf{C})$ to provide a posterior distribution over the true function values given the observed data, and an acquisition function that uses the posterior distribution generated by the surrogate model to provide the guidance of the design space $\mathbf{C}$ to balance exploration and exploitation. *Exploitation* is evaluating in areas where the surrogate model estimates a good objective prediction. However, too much exploitation is a waste of time and resources as the gain may be marginal, as well as the risk of exploiting a local minimum and missing a global minimum. *Exploration* is evaluating in a region that has high uncertainty to prevent the model from being trapped in the local optimum, and thus there exists a higher possibility of finding the global optimum. A Gaussian process (GP) is usually used as the surrogate model because it benefits from properties inherited from the Normal distribution. This allows the distributions of various derived quantities to be obtained explicitly, which subsequently empowers the acquisition function to assign a utility value to a set of candidates to be evaluated on the true expensive-to-evaluate func-

tion. Thus Bayesian optimization uses very little data to optimize the black-box design functions, enabling fast and data-efficient design optimization.

We propose a multi-objective Bayesian Optimization (MOBO) algorithm to adaptively change the keyboard size. First, we initialize the model $f(\mathbf{C})$ by placing a Gaussian process prior on it. Then we generate $n = n_0$ initial candidate points by drawing samples from quasi-Monte Carlo (qMC) sampler, and observe outputs of $f(\mathbf{C})$, which is the performance measures at the initial candidate points. Second, we update the posterior probability distribution on $f(\mathbf{C})$ by fitting past observed data which is the vector composed of the performance metrics and the design parameters describing the candidates to the GP model (Surrogate Model), and define the qMC sampler to estimate and optimize the acquisition function and get N new suggested design candidates (candidates of different layout sizes). Third, we observe new performance metrics $f(\mathbf{C})$ at new design candidates and update training points (past observed data). Fourth, we reinitialize the models so that they are ready for fitting on the data points from the next iteration. Then we repeat step two to four iteratively to form the Pareto front.

We find the candidates on the Pareto front by: 1) computing the convex hull of points; 2) saving the non-dominated points from the convex hull; and 3) filtering the dominated points by the convex hull elements. Finally, we return design candidates on the Pareto front and the user can choose the final candidate from the selection of candidates based on their subjective preferences. This algorithm provides a detailed view of how Bayesian optimization works by iteratively and intelligently collecting data by the acquisition function and fitting the Gaussian process (surrogate model) to the collected data. We describe the implementation of the surrogate model and the acquisition function in detail in Sections 4.1 and 4.2.

4.1 Surrogate Model

We use a GP as the surrogate model. In this GP, we use a standard homoskedastic Gaussian likelihood with inferred noise level:

$$p(y \mid f) = f + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2) \tag{1}$$

where y represents noisy observed function values, f represents noise-free observed function values, and σ is a *noise* parameter. The *noise* parameter is modeled by a Gamma distribution which is parameterized by concentration α and rate β.

We further compute the covariance matrix based on the Matern kernel with Automatic Relevance Discovery (ARD) between inputs $\mathbf{x}$ and $\mathbf{x}'$. Informally, a kernel function furnishes a notion of similarity between points. The Matern covariance function can be seen as a generalization of the Gaussian radial basis function. The Matern kernel is used as we believe the data points to be more correlated if they are closer in the design space. It has been shown to perform well in an application where font is made adaptive [27]. We then apply an output scale on the Matern kernel to make this kernel suitable to our specific problem and dataset:

$$k_{\text{scaled}} = \theta_{\text{scale}} k_{\text{orig}} \tag{2}$$

where θ_{scale} is the output scale parameter.

4.2 Acquisition Function

A natural acquisition function for MOBO is Expected Hypervolume Improvement (EHVI), which is the expected increment of the hypervolume indicator. It has been shown to achieve good convergence and diversity to a true Pareto front [8, 52, 53, 58]. Hypervolume measures the volume of the space dominated by the Pareto set and bounded from below by a reference point. The reference point is the lower bound on the objectives. The lower bound is the minimum acceptable value of interest for each objective. We set the reference point by using domain knowledge so that the reference point is made to be slightly worse than the lower bound of objective values. The Hypervolume is computed by partitioning the space dominated by the Pareto front [34, 51].

As our goal is to optimize two objectives and generate multiple candidate points per design iteration, we need to estimate and optimize $\alpha_{qEHVI}(\mathrm{X}_{cand})$, which is an exact computation of the joint EHVI of q new candidate points, X_{cand} (up to the Monte-Carlo (MC) integration error) [10]. The candidate points are the candidates of different layout sizes to be evaluated on the keyboard. Because the objectives are not independent and modeled with independent GPs, we can not express $\alpha_{qEHVI}(\mathrm{X}_{cand})$ in closed form [52]. In line with prior work [10], we use the re-parameterization trick and quasi-Monte Carlo (qMC) sampling using Sobol sequences for optimization and estimation of the $\alpha_{qEHVI}(\mathrm{X}_{cand})$ acquisition function [49]. More details of the computation of the $\alpha_{qEHVI}(\mathrm{X}_{cand})$ can be found in [10].

4.3 Implementation Details

We implement the MOBO in BoTorch as it provides implementations for a number of acquisition functions specifically for multi-objective scenarios [3]. We leverage several features of BoTorch for GPU acceleration and quasi-second order methods for acquisition optimization to ensure efficient computation and optimization. We impose a Gamma prior on the *noise* parameter with concentration, $\alpha = 1.1$ and rate, $\beta = 0.05$. We further constrain *noise* with a lower bound of 10^{-4} and an initial value of 2, which is calculated from $\frac{\alpha-1}{\beta}$. These practices are in line with prior work [10]. We apply a Gamma prior to the output scale parameter with concentration, $\alpha = 2.0$ and rate, $\beta = 0.15$. The smoothness parameter in the Matern kernel is set to 2.5 and we apply a Gamma prior with concentration 3.0 and rate 6.0 to the length scale parameter. We use 128 as the number of samples in the quasi-MC based sampler. For each iteration, we generate $q = 2$ candidates jointly using 20 random restarts and 1024 raw samples via multi-start optimization.

5 Evaluation

The primary goal of this evaluation is to test whether the Bayesian optimization approach applied to adapting the keyboard size can lead to a performance improvement in comparison with the baseline size (same as the HoloLens 2 system keyboard size), which is held constant. To this end, we carried out a user study where participants completed a text entry task on the mid-air gesture keyboards with various layout sizes.

5.1 Method

We use SHARK^2 [32], a simple but effective gesture keyboard decoding strategy to demonstrate the improvements achievable by introducing personalization. SHARK^2 has been previously implemented by many researchers under various settings and forms [39, 54]. SHARK^2 decodes the gesture trajectories by measuring the similarity through different channels, such as shape and location, between the target trajectories and template trajectories, which simply connect the key centers. SHARK^2 is chosen not only because it is lightweight, but also it is explainable and bias-neutral for different keyboard geometries compared to data-driven decoders, which require a large amount of training data that can induce strong bias [2]. Template matching in SHARK^2 is performed in two dimensions by taking the x and y coordinates in the keyboard coordinate system. This is motivated by users' tendency to gesture consistently within a plane [13]. SHARK^2 returns the top-n words based a ranking of their corresponding probability.

We further use simulated word auto-correction to simulate a robust decoder with high-quality auto-corrections in a similar vein to Dudley et al. [15]. When using the system for personalization, users are not provided with a facility for correcting any errors encountered. Instead, as long as one of the top five ranked word recognitions

Figure 2: Visualization of two different design candidates. The green dot indicates lift on and the red dot indicates lift off. The top left boxes show the participant ID and the design candidate ID respectively.

matches the target word, the output is automatically chosen as the target word. This experimental setting is used to minimize user input noise, such as the time to consider which alternative word to choose and perform word selection on the interface. These noise sources also depend on the design of the word selection interface and therefore we try to minimize the dependent effect of the two objectives on other design elements other than the two parameters we have chosen: the width and the height-to-width ratio. The rationale for using a simulated auto-correction approach is that SHARK^2, despite its easy implementation and suitability for our use-case as a size-independent decoder, is not robust enough to recognize the trajectories gestured by the user in 3D space due to noisy trajectories introduced in mid-air gesturing. Therefore, we simulate a robust decoder to provide an accurate decoding capability, essentially serving as a 'Wizard-of-Oz' [9] decoder for the users. Furthermore, using simulated auto-correction does not suggest that the final accuracy performance is biased. Instead, as long as all the design candidates are using the same decoder, the result will be consistent throughout.

5.2 Participants

We recruited 12 participants (8 male, 4 female, mean age = 22.2, standard deviation = 3.3, max = 27, min = 20).

5.3 Baseline

We use a baseline keyboard which has constant size (the same size as the system keyboard on the HoloLens 2). We chose to implement our own keyboard in order to exercise precise control over the keyboard's appearance and behavior. Before and after the personalization process, participants enter phrases using the baseline keyboard size so that the improvement yielded by MOBO could be assessed. This protocol also serves to identify if there exists a learning effect between the start and the end of the personalization process.

5.4 Stimulus Phrase Set

We used the Enron Mobile Corpus [28] and the MacKenzie phrase set [37] as stimuli. We constrained the length of phrases to 20–60 characters and 4–16 words. The final stimulus set consisted of 1,598 phrases with 2,428 unique words. Each phrase contained on average 7.05 words, with the maximum and the minimum number of words being 15 and 4 respectively.

5.5 Procedure

Participants were first instructed to complete a practice session to develop their gesture typing skills using the experimental application. Keyboard geometries in the practice session were randomly selected to prevent familiarization towards certain geometries and thus biasing the Bayesian optimization process. The reference point for qEHVI optimization is set from the practice session, which is 80% of the average of the aggregated speed and accuracy for three randomly selected design candidates.

Thereafter the actual adaption process began (see Figure 3). This process consisted of two stages. The first stage was MOBO and the

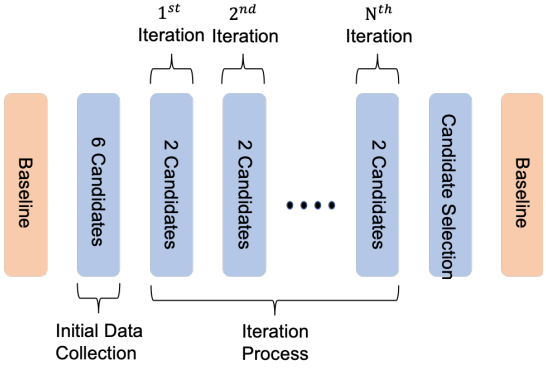


Figure 3: The procedure for the user study.

second stage was to enable the user to choose from the generated Pareto front based on their own subjective preferences, considering the trade-off between speed and accuracy as well as their visual preferences. For each design candidate during the MOBO process and the baseline size, each participant was instructed to gesture 20 phrases of which the first 10 were dropped from the analysis to minimize any initial variability associated with learning a new size. This was the familiarization stage. The choice of dropping 10 phrases was determined from our preliminary results that demonstrated that the performance curve saturated at around 10 phrases, after which the intrinsic performance metrics of the specific design candidate emerged. The accuracy and entry rate of the remaining 10 phrases for each design candidate were collected to update the surrogate model. Furthermore, during each candidate, participants were periodically reminded to pay attention to their accuracy or speed if either accuracy or speed decreased below 10% of the aggregated measures from the familiarization stage.

We used six design candidates as the number of initial samples for MOBO. Thereafter, in the MOBO process, participants started by entering text based on six initial design candidates suggested by the initialized acquisition function. The choice of the number of initial samples was based on an established heuristic for the number of initial samples, $2(p+1)$, where p is the number of varying parameters. Each iteration will suggest two candidates. The number of iterations plays a more important role than the number of data points in each iteration [10]. Therefore, five iterations with two data points per iteration will enable the GP to better fit to the function $f(\mathbf{C})$ than two iterations with five data points. In order to provide the user with a better experience, we should not require users to gesture many different candidates overall because such a process can overly burden users. Each completion of a design candidate is followed by a 10-minute break to ensure that muscle fatigue did not influence the results. After the initialization stage, the acquisition function would suggest two design candidates in each iteration for the participant to complete. The MOBO stage ended when the hypervolume stopped

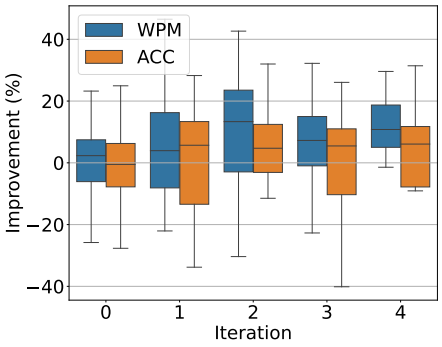


Figure 4: Box plots depicting median and quartiles of the percentage improvements from the baseline candidate for speed and accuracy of each iteration during the optimization process. Iteration 0 is the initial data collection. The improvement on the y-axis indicates the percentage improvement of the performance (speed and accuracy) of the new design candidates at each interaction compared to the baseline candidate.

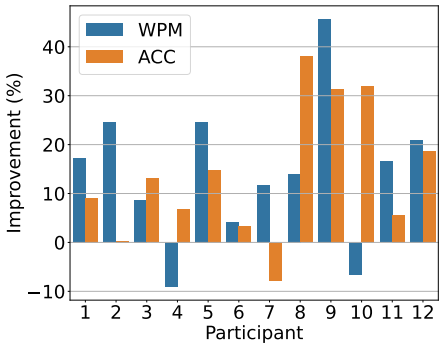


Figure 5: Percentage improvement from the baseline candidate to the optimal candidate for speed and accuracy for the individual participants. The improvement on the y-axis indicates the percentage improvement of the performance (speed and accuracy) of the final selected design candidate compared to the baseline candidate.

increasing for at least two iterations as preliminary experiments suggested that if the hypervolume stopped increasing for more than two iterations, it rarely increased further and the marginal gain was small. When MOBO was completed, a Pareto front was shown to each participant in a gallery view where candidates would be displayed consecutively. Each participant could then select their preferred design from the Pareto front. All the performance metrics were reported as aggregates and reset after the familiarization stage for each design candidate. At the completion of the study, participants were asked to comment on the two questions "What do you think of the personalization process?", and "What do you think of the selected candidates?".

6 Results

Participants selected their preferred choice from the Pareto front determined using the MOBO process and indicated that the designs presented were valid and matched their preferences. Overall, we found that the percentage improvement from the optimal candidate over the baseline was on average 14.4% with a standard deviation of 14.07% for speed and 13.8% with a standard deviation of 13.41% for accuracy. A paired t-test showed that these differences were statistically significant for both speed ($t(11) = -3.40$, $p = 0.0059$) and accuracy ($t(11) = -3.55$, $p = 0.0046$).

To tease out any learning/adaption effects resulting from participants simply becoming better at the task, we took the speed and accuracy differences between the baseline performances before and after the evaluation process from the 12 participants and calculated the mean and standard deviation. The mean and standard deviation for the differences in speed were 0.85 WPM and 1.70 WPM respectively. The mean and standard deviation of the differences in accuracy were -1.32% and 2.61% respectively. Given these small differences we can conclude that any learning/adaptation effect stemming from the participants themselves was negligible.

Figure 7 shows the various final candidates selected from the Pareto front by the individual users. The candidates are represented as dots that have varied normalized height-to-width ratio and normalized width ratio in the design space.

Figure 4 shows the percentage improvement of speed and accuracy for each iteration along the adaptive process. We can see the performance measures increase with the adaptive process. Figure 5 shows the percentage improvement of speed and accuracy from the design candidates that were presented in the Pareto front and ultimately chosen by the participant as their preferred design. We observe that most of the chosen candidates outperformed the baseline candidate in both speed and accuracy. *P9* shows the most benefit from the personalization process with more than 40% improvement on text entry speed. We observed that the width of the *P9* final chosen candidate is 25% smaller than that of the baseline candidate, and the height-to-width ratio is 43% larger than that of the baseline candidate. The exceptions are *P4, P7, P10*, as one of the performance metrics deteriorated. However, the other metric shows an improvement, demonstrating a possible personal preference to achieve high performance on one objective over the other. The baseline condition's size was derived from the HoloLens 2 system keyboard, and it is reasonable to expect that some effort has been taken to ensure the system keyboard design yields an acceptable performance for some users (especially *P4, P7, P10*). Crucially, it is clear that different participants have very different preferences. While some of them prefer a balanced trade-off between speed and accuracy, some of them have a strong preference for one over another, such as *P2* who has chosen the candidate that offers a large speed improvement but negligible accuracy improvement. This reinforces the motivation of this work.

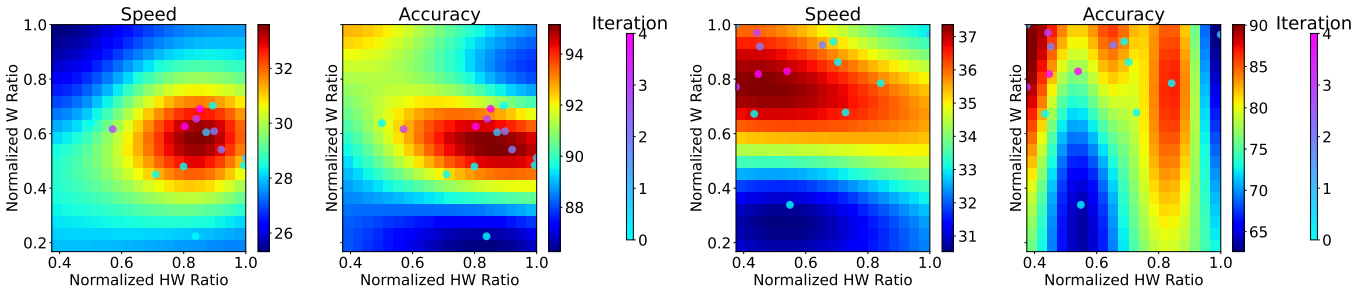
6.1 Qualitative Feedback

In the post-study questionnaire, participants were also asked to provide responses to questions asking what they think of the process and the final candidates. We analyzed the qualitative responses from the participants by grouping related keywords referring to their user experience on the process and on the result: 1) fast, efficient; 2) balance, trade-off; 3) smart, intelligent. These results are provided on the basis that they allow an indication of the individual participants' sentiments in our study. We do not claim these remarks generalize to the wider population. The common feedback that emerged from participants' comments is summarized below:

1. This personalization process is efficient to find the final candidates. (*P4, P5, P8, P9, P10, P12*)

2. The final candidate selection achieves a balance between speed and accuracy. (*P1, P2, P3, P5, P8, P9, P11, P12*)

3. The process is intelligent such that it often gives a better candidate in terms of the trade-off of speed and accuracy at every iteration after the first several candidates. (*P2, P3, P5, P9, P11, P12*)



(a) Participant 1 (*P1*)

(b) Participant 2 (*P2*)

Figure 6: *Left:* Predicted speed (WPM), *Right:* Predicted accuracy (ACC), over the design space for participants *P1* and *P2*. The design space is defined by the space formed by different design candidates of various normalized width (W) ratios and normalized heigh-to-width (HW) ratios. The heat map shows the predicted speed and accuracy of each individual candidate in the design space. Progression of the MOBO process is indicated by the points using the *Iteration* color scale.

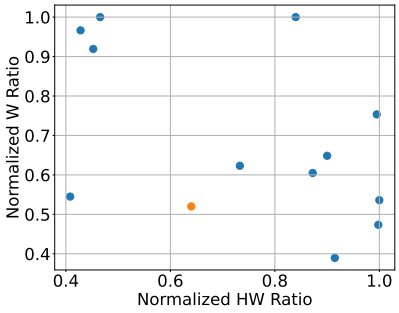


Figure 7: The blue dots represent the various final design candidates chosen by the participants from the Pareto front. The orange dot represents the baseline candidate's position in the design space. This figure summarizes the final candidates chosen by all participants and highlights the significant variety in the selected designs. This result indicates that keyboard size affects each user differently and it is difficult to arrive at any generalizable relationship.

Specifically, for responses regarding final candidate performance, *P8* commented, "I really like the final candidates that it suggested to me, all of them seems to be one of the most comfortable keyboard layouts I tried during the personalization process." Furthermore, participants also commented on the optimization process efficiency. For example, *P9* commented, "The personalization process is efficient and I quickly reached to the final candidate." However, two participants, *P7* and *P10*, commented that the personalization process was tedious despite the fact that the final candidates found suited their preferences for a speed and accuracy balance. Overall, the participants in the study were positive in their feedback on usability in terms of the efficiency of the personalization and noted the final selected candidates effectively balanced performance in terms of speed and accuracy.

7 Discussion

The results from the user study demonstrate the effectiveness of the MOBO process for the adaptation of size and reveal the very different behaviors exhibited by users, motivating the need for personalization of mid-air gesture keyboards. The MOBO process is validated through the gains in speed and accuracy we observed in the user study. Overall, the MOBO process is efficient and effective as our results illustrate an average 14.4% gain for speed and a 13.8% gain for accuracy after only four iterations, and there are only two candidates per iteration.

7.1 Personalization is Important

The plots in Figure 6 show the various mappings from the design space to the objective space for two participants. Due to the limited space in the paper we do not present the plots for all participants but we observe that different participants have distinctive characteristics of mappings from the design space to the objective space. This emphasizes the importance of building adaptive user interfaces for mid-air virtual keyboards due to different user behavior, mental and physiological models. Further, we observe in Figure 7 that the final candidates, which exhibit great variation, reveal that various candidates suit individual users. This strengthens the aforementioned argument on the importance of having a keyboard personalized through an adaptive process. We conjecture that this type of distinct inter-user variation in behavior may also exist in other interactions in AR. Adaptive user interface design could therefore be fruitful in optimizing other AR interactions for individuals.

7.2 Streamlining Personalization with AdaptiKeyboard

The experimental procedure we employed simulates daily usage of a mid-air gesture keyboard collapsed into a single experimental session. In practice, we expect that a user will be exposed to approximately 14 layout sizes as part of the personalization process over several days. This is because the Bayesian optimization procedure requires 6 candidates at the initialization stage and then approximately 4 iterations (2 candidates each) to reliably identify the Pareto front for this multi-objective optimization problem.

Assume an average user enters text at a mean rate of 40 sentences a day. Since we use 20 sentences to assess one layout size this means two layout sizes can be tested per day. Users are required to enter text presented to them as stimuli from a phrase set for the purpose of the personalization. Each day requires two candidates from the users (approximately 5 minutes per candidate if we assume the average text entry speed is 20 WPM). Therefore, a total of 14 layout sizes can be assessed within a week without excessively burdening the user. After a week's usage, the Pareto efficient designs are identified and displayed to the user. The user can choose from among these suggested alternatives based on the reported performance and thereby incorporate their own personal preferences.

More iterations (more data for fitting the GP model) are likely to improve the model's prediction of the output of the function $f(\mathbf{C})$ and thus help identify better designs in terms of speed and accuracy. However, if this personalization process takes too long, users will face a high cognitive burden due to the constantly changing keyboard size.

8 Limitations and Future Work

The optimization algorithm proposed in this paper is specifically developed for the task of personalizing a mid-air gesture keyboard. There is no suitable existing algorithm that has been proposed to suit this particular task. Moreover, the main focus of the paper is to demonstrate that a mid-air gesture keyboard can be effectively and efficiently personalized through MOBO. We therefore did not introduce another optimization algorithm as a baseline and leave such a comparison for future work.

In reality, users adapt their behavior both consciously and unconsciously as they use an interface. Therefore, a user interacting with an adaptive interface forms an interesting feedback loop in which both the user and the system co-adapt. Despite our results from the baseline performances before and after the evaluation process suggesting that users did not adapt much to the system, we still see promising future work in carefully studying the role of human adaptivity in system adaptivity within the realm of intelligent text entry in AR.

While this paper serves as a proof of concept for interactive utility, our system still lacks some fundamental features required of any practical text input system, such as error correction, word suggestions or punctuation. Moreover, despite the fact that the current personalization procedure is reasonably efficient as it only takes 14 days with 10 minutes per day to complete, the procedure is nonetheless tedious as users need to enter text as prompted by stimuli. We see promising future work in exploring means of enabling a personalization procedure that is less demanding on users so that the procedure can smoothly blend into their daily usage of the mid-air gesture keyboard. However, we caution that this is highly challenging in practice as it will be difficult to accurately assess users' speed when users are composing text as opposed to transcribing stimulus sentences.

Another potential avenue of future work is to enable users to determine the size of the keyboard on their own. This would allow a comparison of user-led personalization with algorithmic-lead personalization.

9 Conclusions

In this paper, we have demonstrated an effective technical method for designing an adaptive mid-air gesture keyboard using multi-objective Bayesian optimization. We have made the keyboard size—the width and aspect ratio—adaptive to users' behavior. We used MOBO to identify an optimized keyboard size that balances speed and accuracy for each individual. We have shown that this approach can deliver a statistically significant 14.4% and a 13.8% improvement in speed and accuracy respectively relative to a baseline design derived from the system gesture keyboard on the HoloLens 2. The process presented in this paper has demonstrated tangible gains on mid-air gesture keyboard performance by adapting to individuals' performance and preferences. We believe personalization will be key to tackle the dynamic constraints that arise due to the nature of hand-tracked mid-air gesture typing using optical see-through head-mounted displays. We believe promising future work includes exploring how to include users further in the personalization loop by explaining to users the nature of adaptivity and finding means of transferring agency of the adaptive process from the system to the user.

Acknowledgments

Complete source code for the proposed multi-objective Bayesian optimization algorithm in performing the mid-air gesture keyboard adaptation introduced in this paper is available as supplemental materials. John Dudley and Per Ola Kristensson were supported by the EPSRC (grant EP/S027432/1).

References

[1] Hololens 2 release notes. `https://docs.microsoft.com/en-us/hololens/hololens-release-notes#swipe-to-type`. Microsoft.

[2] O. Alsharif, T. Ouyang, F. Beaufays, S. Zhai, T. Breuel, and J. Schalkwyk. Long short term memory neural network for keyboard gesture decoding. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2076–2080, 2015.

[3] M. Balandat, B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy. BoTorch: A framework for efficient Monte-Carlo Bayesian optimization. *Advances in neural information processing systems*, 33, 2020.

[4] D. Benyon and D. Murray. Applying user modeling to human-computer interaction design. *Artificial Intelligence Review*, 7(3):199–225, 1993.

[5] X. Bi and S. Zhai. IJqwerty: What difference does one key change make? gesture typing keyboard optimization bounded by one key position change from qwerty. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 49–58, 2016.

[6] X. Cao and S. Zhai. Modeling human performance of pen stroke gestures. CHI '07, p. 1495–1504. Association for Computing Machinery, New York, NY, USA, 2007.

[7] L. Chan, Y.-C. Liao, G. B. Mo, J. J. Dudley, C.-L. Cheng, P. O. Kristensson, and A. Oulasvirta. Investigating positive and negative qualities of human-in-the-loop optimization for designing interaction techniques. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 2022.

[8] I. Couckuyt, D. Deschrijver, and T. Dhaene. Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. *Journal of Global Optimization*, 60(3):575–594, 2014.

[9] N. Dahlbäck, A. Jönsson, and L. Ahrenberg. Wizard of Oz studies—why and how. *Knowledge-based systems*, 6(4):258–266, 1993.

[10] S. Daulton, M. Balandat, and E. Bakshy. Differentiable expected hypervolume improvement for parallel multi-objective Bayesian optimization. *arXiv preprint arXiv:2006.05078*, 2020.

[11] P. David. Clio and the economics of qwerty. *The American Economic Review*, 75:332–7, 05 1985.

[12] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.

[13] N. K. Dim, C. Silpasuwanchai, S. Sarcar, and X. Ren. Designing mid-air tv gestures for blind people using user- and choice-based elicitation approaches. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*, DIS '16, p. 204–214. Association for Computing Machinery, New York, NY, USA, 2016. doi: 10.1145/2901790.2901834

[14] T. Drey, P. Jansen, F. Fischbach, J. Frommel, and E. Rukzio. Towards progress assessment for adaptive hints in educational virtual reality games. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, pp. 1–9, 2020.

[15] J. Dudley, H. Benko, D. Wigdor, and P. O. Kristensson. Performance envelopes of virtual keyboard text input strategies in virtual reality. In *IEEE International Symposium on Mixed and Augmented Reality*, pp. 289–300, 2019.

[16] J. J. Dudley, J. T. Jacques, and P. O. Kristensson. Crowdsourcing interface feature design with Bayesian optimization. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2019.

[17] J. J. Dudley, J. T. Jacques, and P. O. Kristensson. Crowdsourcing design guidance for contextual adaptation of text content in augmented reality. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2021.

[18] J. J. Dudley, K. Vertanen, and P. O. Kristensson. Fast and precise touch-based text entry for head-mounted augmented reality with variable occlusion. *ACM Trans. Comput.-Hum. Interact.*, 25(6), Dec. 2018.

[19] M. Dunlop and J. Levine. Multidimensional Pareto optimization of touchscreen keyboards for speed, familiarity and improved spell checking. In *Proceedings of the SIGCHI Conference on Human Factors in*

Computing Systems, p. 2669–2678, 2012.

[20] J. a. M. Evangelista Belo, A. M. Feit, T. Feuchtner, and K. Grønbæk. XRgonomics: Facilitating the creation of ergonomic 3D interfaces. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2021.

[21] L. Findlater and K. Z. Gajos. Design space and evaluation challenges of adaptive graphical user interfaces. *AI Magazine*, 30(4):68–68, 2009.

[22] P. I. Frazier. A tutorial on Bayesian optimization. *arXiv preprint arXiv:1807.02811*, 2018.

[23] K. Z. Gajos, D. S. Weld, and J. O. Wobbrock. Automatically generating personalized user interfaces with supple. *Artificial Intelligence*, 174(12-13):910–950, 2010.

[24] S. Greenhill, S. Rana, S. Gupta, P. Vellanki, and S. Venkatesh. Bayesian optimization for adaptive experimental design: A review. *IEEE access*, 8:13937–13948, 2020.

[25] J. Hussain, A. U. Hassan, H. S. M. Bilal, R. Ali, M. Afzal, S. Hussain, J. Bang, O. Banos, and S. Lee. Model-based adaptive user interface based on context and user experience evaluation. *Journal on Multimodal User Interfaces*, 12(1):1–16, 2018.

[26] J. Jokinen, A. Acharya, M. Uzair, X. Jiang, and A. Oulasvirta. Touchscreen typing as optimal supervisory control. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2021.

[27] F. Kadner, Y. Keller, and C. Rothkopf. AdaptiFont: Increasing individuals' reading speed with a generative font model and Bayesian optimization. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–11, 2021.

[28] B. Klimt and Y. Yang. The Enron corpus: A new dataset for email classification research. In J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, eds., *Machine Learning: ECML 2004*, pp. 217–226. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.

[29] P. O. Kristensson. *Discrete and Continuous Shape Writing for Text Entry and Control*. PhD thesis, 2007.

[30] P. O. Kristensson. Five challenges for intelligent text entry methods. *AI Magazine*, 30(4):85–85, 2009.

[31] P. O. Kristensson. Next-generation text entry. *Computer*, 48(07):84–87, 2015.

[32] P. O. Kristensson and S. Zhai. SHARK^2:a large vocabulary shorthand writing system for pen-based computers. *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, 01 2004.

[33] K. Kurzhals, F. Göbel, K. Angerbauer, M. Sedlmair, and M. Raubal. A view on the viewer: Gaze-adaptive captions for videos. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2020.

[34] R. Lacour, K. Klamroth, and C. Fonseca. A box decomposition algorithm to compute the hypervolume indicator. *Computers & Operations Research*, 79, 07 2016.

[35] D. Lindlbauer, A. M. Feit, and O. Hilliges. Context-aware online adaptation of mixed reality interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pp. 147–160, 2019.

[36] M. N. Lystbæk, K. Pfeuffer, J. E. S. Grønbæk, and H. Gellersen. Exploring gaze for assisting freehand selection-based text entry in a;r. *Proc. ACM Hum.-Comput. Interact.*, 6(ETRA), may 2022.

[37] I. S. MacKenzie and R. W. Soukoreff. Phrase sets for evaluating text entry techniques. In *CHI '03 Extended Abstracts on Human Factors in Computing Systems*, p. 754–755. Association for Computing Machinery, New York, NY, USA, 2003.

[38] I. S. MacKenzie and S. X. Zhang. The design and evaluation of a high-performance soft keyboard. CHI '99, p. 25–31. Association for Computing Machinery, New York, NY, USA, 1999.

[39] A. Markussen, M. R. Jakobsen, and K. Hornbæk. Vulture: A mid-air word-gesture keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 1073–1082, 2014.

[40] R. Oppermann. Adaptively supported adaptability. *International Journal of Human-Computer Studies*, 40(3):455–472, 1994.

[41] S. Park, C. Gebhardt, R. Rädle, A. M. Feit, H. Vrzakova, N. R. Dayama, H.-S. Yeo, C. N. Klokmose, A. Quigley, A. Oulasvirta, et al. Adam: Adapting multi-user interfaces for collaborative environments in real-time. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2018.

[42] L. Purvis, S. Harrington, B. O'Sullivan, and E. C. Freuder. Creating personalized documents: an optimization approach. In *Proceedings of the 2003 ACM symposium on Document engineering*, pp. 68–77, 2003.

[43] S. Reyal, S. Zhai, and P. O. Kristensson. Performance and user experience of touchscreen and gesture keyboards in a lab setting and in the wild. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 679–688, 2015.

[44] J. Shen, J. Dudley, and P. O. Kristensson. Simulating realistic human motion trajectories of mid-air gesture typing. In *2021 IEEE International Symposium on Mixed and Augmented Reality*, pp. 393–402, 2021.

[45] B. A. Smith, X. Bi, and S. Zhai. Optimizing touchscreen keyboards for gesture typing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 3365–3374, 2015.

[46] C. Stephanidis, A. Paramythis, M. Sfyrakis, A. Stergiou, N. Maou, A. Leventis, G. Paparoulis, and C. Karagiannidis. Adaptable and adaptive user interfaces for disabled users in the avanti project. In S. Trigila, A. Mullery, M. Campolargo, H. Vanderstraeten, and M. Mampaey, eds., *Intelligence in Services and Networks: Technology for Ubiquitous Telecom Services*, pp. 153–166. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[47] K. Todi, G. Bailly, L. Leiva, and A. Oulasvirta. Adapting user interfaces with model-based reinforcement learning. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, pp. 1–13, 2021.

[48] K. Todi, D. Weir, and A. Oulasvirta. Sketchplore: Sketch and explore with a layout optimiser. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems*, pp. 543–555, 2016.

[49] J. T. Wilson, R. Moriconi, F. Hutter, and M. P. Deisenroth. The reparameterization trick for acquisition functions. *arXiv preprint arXiv:1712.00424*, 2017.

[50] W. Xu, H. Liang, A. He, and Z. Wang. Pointing and selection methods for text entry in augmented reality head mounted displays. In *2019 IEEE International Symposium on Mixed and Augmented Reality*, pp. 279–288, 2019.

[51] K. Yang, M. Emmerich, A. Deutz, and T. Bäck. Efficient computation of expected hypervolume improvement using box decomposition algorithms. *Journal of Global Optimization*, 75(1):3–34, 2019.

[52] K. Yang, M. Emmerich, A. Deutz, and T. Bäck. Multi-objective Bayesian global optimization using expected hypervolume improvement gradient. *Swarm and evolutionary computation*, 44:945–956, 2019.

[53] K. Yang, D. Gaida, T. Bäck, and M. Emmerich. Expected hypervolume improvement algorithm for pid controller tuning and the multiobjective dynamical control of a biogas plant. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1934–1942. IEEE, 2015.

[54] C. Yu, Y. Gu, Z. Yang, X. Yi, H. Luo, and Y. Shi. Tap, dwell or gesture? exploring head-based text entry techniques for hmds. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, p. 4479–4488, 2017.

[55] S. Zhai, M. Hunter, and B. A. Smith. The metropolis keyboard-an exploration of quantitative techniques for virtual keyboard design. In *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pp. 119–128, 2000.

[56] S. Zhai and P.-O. Kristensson. Shorthand writing on stylus keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, p. 97–104. Association for Computing Machinery, New York, NY, USA, 2003.

[57] M. R. Zhang, S. Zhai, and J. O. Wobbrock. Text entry throughput: Towards unifying speed and accuracy in a single performance metric. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, p. 1–13. Association for Computing Machinery, New York, NY, USA, 2019.

[58] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Transactions on evolutionary computation*, 7(2):117–132, 2003.