Touchscreen Typing As Optimal Supervisory Control

Jussi P.P. Jokinen
Department of Communications and Networking, Aalto University, Finland

Aditya Acharya
Department of Computer Science, Aalto University, Finland

Mohammad Uzair
Department of Communications and Networking, Aalto University, Finland

Xinhui Jiang
Center for Human-Engaged Computing (CHEC), Kochi University of Technology, Japan

Antti Oulasvirta
Department of Communications and Networking, Aalto University, Finland

ABSTRACT

Traditionally, touchscreen typing has been studied in terms of motor performance. However, recent research has exposed a decisive role of visual attention being shared between the keyboard and the text area. Strategies for this are known to adapt to the task, design, and user. In this paper, we propose a unifying account of touchscreen typing, regarding it as optimal supervisory control. Under this theory, rules for controlling visuo-motor resources are learned via exploration in pursuit of maximal typing performance. The paper outlines the control problem and explains how visual and motor limitations affect it. We then present a model, implemented via reinforcement learning, that simulates co-ordination of eye and finger movements. Comparison with human data affirms that the model creates realistic finger- and eye-movement patterns and shows human-like adaptation. We demonstrate the model's utility for interface development in evaluating touchscreen keyboard designs.

CCS CONCEPTS

• **Human-centered computing → HCI theory, concepts and models**;

KEYWORDS

touchscreen typing, computational modelling, rational adaptation

ACM Reference Format:
Jussi P.P. Jokinen, Aditya Acharya, Mohammad Uzair, Xinhui Jiang, and Antti Oulasvirta. 2021. Touchscreen Typing As Optimal Supervisory Control. In *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3411764.3445483

1 INTRODUCTION

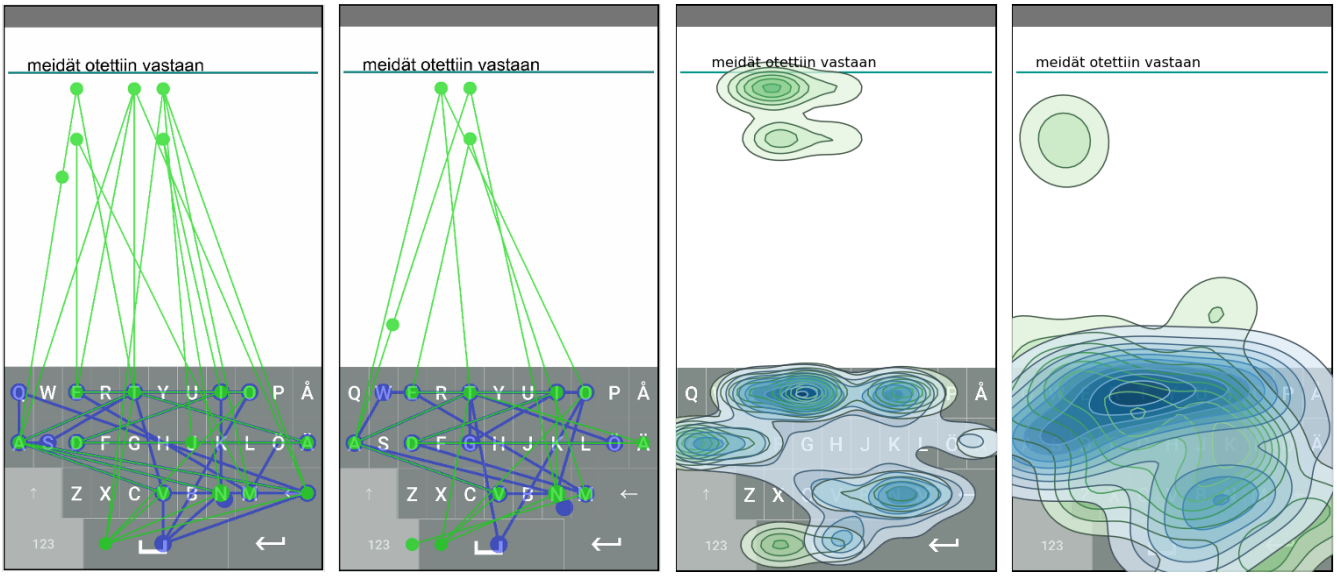
Motor performance has been central to HCI research into touchscreen typing for two decades. However, recently the view has been complicated by evidence of another mechanism, much less understood. In close analysis of how users move their eyes and fingers while typing, *gaze deployment* was discovered to be strongly associated with typing performance [31]. In essence, a 'resource conflict' appears to exist between the subtasks of typing and proofreading, both requiring visual attention for efficient performance. Fast typists can minimise attention shifts, whereas slower ones must jump between text display and keyboard, thereby hampering performance. Furthermore, it seems that, in general, no single optimal eye–hand strategy exists. Rather, strategy was found to *adapt* as a function of error rate and skill level. A question stands out: what underpins this adaptive ability? Our paper presents the first step toward a unifying account of typing on touchscreen devices. We consider how factors describing the user, task, and design jointly affect how the gaze and fingers are deployed, and we provide a well-grounded novel explanation of the dynamics of typing performance.

Supervisory control refers to the problem of deciding on sharing of limited resources for on-going tasks [19, 43]. *Optimal supervisory control* is the concept that the control decisions for sharing resources or performing actions are taken optimally in terms of some objective, with optimality understood as bounded by the constraints of the agent [11, 23]. The control problem we face in typing is to decide how to deploy visual attention in concert with motor movements [31, 32]. Were we able to perceive, remember, and move perfectly without noise and uncertainty, there would be no need for this. Because the finger movements are noisy, however, visual attention is needed for guiding them. Concurrently, we need vision to monitor our typing progress and to detect typing errors. With our foveated vision, only a small portion of the visual field is seen accurately, ruling out simultaneous finger guidance and error-checking. Touchscreens, with their lack of tactile feedback from the keys, exacerbate the problem. Moreover, the typed text must be held in working memory, with a reference to the word being typed and its characters, whose locations must be reliably and quickly retrieved from long-term memory [34]. It is astonishing how well a skilled typist can orchestrate these subtasks, with high performance in the absence of deliberate practice, even surpassing 80 words per minute (WPM) on a mobile device [47].

With this paper, we develop the theory of optimal supervisory control in the context of touchscreen transcription typing and present a computational model to compare its predictions against human data. Figure 1 presents finger and eye trajectories over the course of typing of one sentence. This demonstrates our model's detailed predictions. A key assumption in our approach is that the supervisor's decisions are optimal, *given its bounds and those of the design.* The model has learnt to shift gaze from the keyboard to the text-entry area to conduct proofreading from time to time,

CHI '21, May 8–13, 2021, Yokohama, Japan

ACM ISBN 978-1-4503-8096-6/21/05.
https://doi.org/10.1145/3411764.3445483



(a) Model typing, with some errors and no automatic error-correction

(b) Model typing, with no errors and with automatic error-correction

(c) Model typing, no errors and no automatic error-correction

(d) Human typing, no errors and no automatic error-correction

Figure 1: Our computational model of optimal supervisory control predicts patterns of finger (blue) and eye (green) movements over a touchscreen. The model – as humans do – can adapt to changes in typing conditions, such as the presence of typing errors (a) or introduction of intelligent error-correction (b). Panes c and d present heatmaps of finger and eye movements for typing of the same sentence by the model and a human. The image in pane d is reprinted from Jiang et al. [31]'s Figure 1, with permission, CC-BY-SA.

this frequency depending on the momentary events of the task, such as making of errors. A hallmark of human behaviour is its remarkable adaptivity to the ecologies of widely different tasks while being constrained by the available cognitive and external resources. This is evident in typing, where users can adjust the way they move and share attention in line with changing conditions, such as varying key sizes or intelligent typing aids. The adjustment is illustrated in Figure 1b, which shows our model's behaviour with intelligent error-correction enabled. Fewer proofreading glances are necessary, because the model has adapted to the device's reliable error correction. In an optimal-control model, these kinds of behaviour adaptations are approximated by computing rules of action that are constrained by noisy motor movements and limited visual acuity. This approach explains typing performance for given user abilities, the typing task, and device specifics. Further, it can be used to explain adaptive changes in behaviour as said factors change. In contrast, the prevailing models of motor control in typing have assumed a fixed strategy, with any change in conditions necessitating the model's re-parameterisation with empirical data [10, 13, 54]. Comparison of our model's predictions (Figure 1c) with human data (Figure 1d) illustrates how our assumptions result in reproducing human-like typing behaviour.

Our work complements earlier typing models and improves on the articulation of the supervisory control problems in HCI. The applicability and extendability of mathematical models, such as Fitts' law, make them popular, but they fail to account for touchscreen typing's complex dynamics and the supervisory control strategy required for efficiently managing those dynamics. They are silent on how vision is directed to the various portions of the keyboard and how visual attention ties in with making, detecting, and correcting errors. More complicated process models, such as a keystroke-level model (KLM) [27, 54] or ACT-R [13], can describe some of these dynamics but seldom can predict *how* behaviour strategies adapt to changes in the task environment. Our model is the first to describe touchscreen typing as an optimal supervisory control problem. The approach renders it able to simulate typing behaviour's constant dynamic adaptation to what happens during the task while also predicting the user's finger and eyes movements over the device during typing. Presently, we have validated our model with only one dataset, covering typing on a single device, but the model is theoretically and technically suitable as a general tool for anticipating how users may adapt to different keyboard layouts and for characterising how the dynamics of typing hinge on the other details of the typing task.

2 RELATED WORK AND GOALS

We begin by reviewing fundamental features of touchscreen typing, with regard to both performance metrics and other known phenomena. Because one assumption essential to our theory of optimal supervisory control is that optimality is bounded, the review of these features is constructed in terms of what constrains optimal control of touchscreen typing. The goal is to establish a set

of features that a realistic model for simulating touchscreen typing should cover.

2.1 Typing on Touchscreen Devices

Human motor control is noisy, creating imprecise movements and uncertainty about the current finger position [24, 44]. Noisy pointing movements may result in typing errors, especially with larger movement arcs. Furthermore, physical motor limitations dictate how quickly the finger can travel, given a key. Similarly, the information processing capacity of human vision is limited [14, 50, 51]. As only a small foveated portion of the full visual field is seen clearly, the area of high visual acuity around a *fixation* must regain focus at times via a *saccade*, a fast ballistic eye movement. Foveated vision, alongside noisy motor movements and the concomitant need for visual guidance of the finger, means that the eye must make continuous movements, some aligned with finger trajectory, during touchscreen typing [31]. Below, we review key metrics and phenomena connected with touchscreen typing and explain them with regard to bounding factors such as noise, uncertainty, and time demands.

2.1.1 Typing speed. One basic speed measurement in typing is the inter-key interval (IKI), the time between consecutive keypresses [21]. With physical keyboards, where typing employs multiple fingers, which can move largely without vision-based guidance, IKI averages of around 170 ms have been recorded [21]; touchscreen typing exhibits much longer IKIs, 380 ms for one-finger typing and 270 ms for two thumbs [31]. The most widely used method for assessing text-entry performance [62] uses words per minute, standardised in units of five characters of typed text divided by typing time [21]. Reports on studies with touchscreen devices generally cite average WPM values between 25 and 40, but figures vary with the typing conditions. Typing with two thumbs is faster than typing with one finger and permits users to reach 40 WPM reliably [7, 12, 31, 42], sometimes even exceeding this [47].

The classic way to model pointing time in HCI is via Fitts' law [22], which predicts the movement time required to reach a target of a certain width that is at a set distance. It and all its variants can be employed to predict typing on touchscreens in terms of IKI, as long as the empirical parameters are correctly calibrated [10]. Fitts' law can be used to predict WPM also, if one assumes that the finger simply moves from one key to the next until the correct character sequence has been entered [56, 61]. As this modelling approach considers primarily the finger movements, in some cases also addressing other operations (such as mental acts required for said pointing [27]), it is suitable for predicting the upper limits of experts' behaviour. The limit thus derived is somewhere around 35 WPM for one-finger and 60 WPM for two-finger typing [7, 39]. Again, these predictions consider typing only as sequential keypresses, disregarding such factors as the possibility of typing errors and their implications for the typing process – e.g., any need for proofreading and correction of errors.

2.1.2 Typing errors. Typing errors are a consequence of noise associated with motor movements. Generally, error rates are higher in typing on a touchscreen keyboard (7–10.8% [7]) than with a physical keyboard (between 0.47% and 0.76% [21]), across various touchscreen devices [6, 60]. Although typing with two thumbs on mobile devices is faster than using one finger, it leads to more errors [7, 31, 42]. Furthermore, it is useful to distinguish between immediate backspacing, to rectify an error that is immediately evident, and delayed backspacing, over multiple characters, when the error is noticed only later [4, 31]. Whilst models based on Fitts' law predict only error-free WPM values, some more recent work has considered the role of typing errors [5, 8, 32, 54], generally in efforts to predict both the circumstances that lead to errors and the errors' impact with regard to the error-correction process [4]. On account of the stochastic nature of making errors, thoroughly modelling error-correction behaviour requires simulations, covering predictions of errors, their detection during proofreading, and how they are corrected, with a clear mechanism for the monitoring role of vision throughout. To our knowledge, only one pre-existing model meets this requirement [54], but it still does not dynamically adapt to making of errors. Neither does it consider changes in typing conditions.

The balance between finger speed and accuracy has a major impact on typing performance. For instance, the number of errors can be considerably reduced by requiring typists to stress accuracy rather than typing speed [63]. This makes typing an optimisation problem: what is the most efficient combination of finger speed and proofreading frequency [32]? Minimising finger-movement time yields faster typing but also results in more errors, thereby demanding more frequent proofreading and compromising speed. Another factor is that incorporating intelligent error-correction can afford faster finger movements without vastly increasing the proofreading effort [9]. Although mathematical models can express the speed–accuracy trade-off and, for instance, predict how typing speed follows changes in key size, there have been no systematic theoretical efforts to determine how optimal speed can be ascertained from the circumstances of the task and the typist's abilities.

2.1.3 Sharing of attention. Users must selectively shift attention among the important areas of their device to overcome the limits imposed by foveated vision. The lack of tactile feedback from the keys, coupled with uncertainty due to noisy pointing movements, necessitates visual guidance of the fingers, so a large proportion of the fixations in touchscreen typing land on the keyboard area. In a recent study, users kept their eyes on the keyboard 60% of the time with two thumbs and 70% of the time with one finger, with occasional glances at the text-entry area for proofreading (on average, 3–4 gaze shifts per sentence) [31].

The only way to realistically model the dynamic coupling of vision and finger movement is via step-by-step simulation of their movements. Production-rule-based models, such as EPIC [38] and ACT-R [2], can do this. These architectures are based on sequential stepping of a simulator, which integrates mathematical formulae that govern how much time each step takes. For instance, a recently developed touchscreen typing model based on ACT-R includes instructions, written by the researcher, for moving the eyes toward a target and following this movement with the finger, for typing-related predictions that consider both eye- and finger-movement time [13]. That model still fails to consider errors and, therefore, also forgoes simulating proofreading, unlike one more advanced, loosely KLM-based model that is used in ability-based optimisation of

Table 1: A comparison of touchscreen typing models, based on the features of typing that they are able to simulate: Fitts' law, ACT-R [13], the ability-based model [54], and the optimal-control model presented in this paper, of which we conclude that only the last can satisfactorily replicate all effects (full circle), due to its rootedness in the theory of optimal supervisory control; other models' failure (empty circle) or partial failing (half full circle) is due to them not considering touchscreen typing to be adaptive behaviour.

Metric or phenomenon	Fitts' law	ACT-R	Ability-based	Optimal control
Average inter-key interval (IKI) and typing speed (WPM)	●	●	●	●
Frequency of errors	◗	◗	●	●
Correlation between error frequency and WPM	◗	◗	●	●
Number of error corrections made	○	○	●	●
Proofreading: frequency and time expended	○	○	◗	●
Amount of immediate vs. delayed error correction	○	○	◗	●
Correlation between error and proofreading frequency	○	○	○	●
Parallel movement of the fingers and eyes	○	○	○	●
Impact of intelligent text entry on typing	○	○	○	●

touchscreens [54]. Neither, however, simulates the parallel motion of finger and eye displayed in human behaviour, wherein the fingers start their movement ahead of the vision in anticipation of the faster saccadic movement that brings the eyes close to the target [31].

Importantly, how attention is shared between the keyboard and the text-entry area is not a global constant or even a purely user-specific one. Rather, it varies between individual instances of typing a message. For instance, there is a positive correlation between the number of corrected typing errors and gaze shifts to the text-entry area [31]. This is because a typist suspecting an error will proofread the typed text and, if necessary, initiate error-correction procedures. Various aspects of noisy finger movements cause some episodes of typing a message to involve more errors than others, resulting in greater need for proofreading. The only model thus far to have simulated proofreading is the above-mentioned ability-based one [54]. That said, it still displays a crucial shortcoming by assuming a constant number of keypresses before the eyes move to the text area, so it does not replicate the dynamic nature of the error-making, detection, and correction phenomena that humans display.

2.2 The Goals for This Paper

Proceeding from our review of both modelling efforts and the key metrics and phenomena of touchscreen typing, summarised in Table 1, we set out to develop and implement a theoretical model of optimal supervisory control. The goal behind this paper is to provide a unified theory of touchscreen typing and, on that basis, implement a model that can realistically simulate all the effects listed in the table. The main purpose of this enterprise is to rectify the lack of a unified theoretical account for how humans are able to discover and execute efficient control programs for tasks that involve noise and uncertainty. Touchscreen text entry is an excellent focus for such theorising, since there are different 'task areas' (the keyboard for typing and the text-entry area for proofreading), both with associated uncertainty. Alongside providing theoretical elucidation, our modelling efforts serve a practical purpose: to support design of touchscreen typing layouts. A model developed to this end should be able to predict how changes in layout, such as adjusting keys' size and positioning, influence various metrics for typing. To address these two goals, the theoretical and the practical one, we 1) present the formal theory of touchscreen typing as optimal supervisory control; 2) evaluate our model with human data, examining its ability to meet the requirements in Table 1; and 3) report on an experiment in which our model demonstrated its ability to adapt its control strategies to intelligent error-correction.

3 OPTIMAL SUPERVISORY CONTROL

A crucial element in our theory is *agency*, the ability to choose those of the available actions that result in desired outcomes. By specifying our model as an agent, we focus on momentary, sequential choices that typists face when interacting with touchscreens. The foundation for our analysis of the agent's choice is *rationality*: from the possible actions, the agent is assumed to select the one that is of the greatest long-term benefit to it. It has been shown that, in response to the complexity of real-world problems, humans adopt *hierarchical* schemes for learning and planning [11, 23]. A hierarchical arrangement renders complex tasks computationally (and cognitively) tractable, as the subtask problems have smaller search spaces. Following these lines, we theorise that human supervisory control is hierarchically organised, with a supervisor dictating how resources are allocated across subtasks and controlling them in other ways. Further, we stipulate that this supervisory control is optimal within the constraints of cognitive capacity and uncertainty, employing a policy that leads to efficient sharing of resources among the subtasks.

The technique of modelling user behaviour and interactions by using the optimal-control framework has surged in popularity in HCI [1, 15, 20, 41, 45, 59]. Such modelling has already been used to explain people's adaptive decision-making behaviour during interaction with data-visualisation systems [17] and for hierarchical menu design [16]. Also, some cognitive models have been applied to tackle the problem of adaptation in complex multidimensional strategy spaces found in co-ordination of perceptual/motor control

[30, 52, 57]. Nevertheless, we are aware of only one previous attempt to apply the idea of optimal supervisory control for simulation of complex and adaptive real-world task behaviour (i.e., driving) [33]. What follows is a detailed description of how to apply this idea formally to model touchscreen typing.

3.1 Optimal Control of Sequential Processes

We describe each subtask individually in terms of a Markov decision process (MDP) [58]. It is a tuple $< S, A, T, R, \gamma >$ whereby sequential problem-solving is formulated as a finite set of states (S), a finite set of actions (A), environmental transition dynamics (T), a reward function (R), and a discount factor (γ). Furthermore, in environments of partial observability, this formalism can be augmented to what is called a partially observable MDP (POMDP), adding to the tuple a finite set of observations (Ω) and an observation function (O) [29, 36]. An agent described via an MDP can take an action $a \in A$ to interact with its environment, potentially causing the environment's state to change from $s \in S$ to new state $s' \in S$ with a probability $T(s, a, s') = p(s' \mid s, a)$. In the case of POMDP, the agent cannot fully observe the state: after performing an action $a \in A$, an observation $o \in \Omega$ is made on the basis of probability $O(s', a, o) = p(o \mid s', a)$.

The reward function R specifies the probability $R(s, a) = p(r \mid s, a)$ of receiving a scalar reward $r \in \mathbb{R}$ after the agent has performed an action and the environment has transitioned. We assume that the agent acts rationally and attempts to maximise its long-term reward. It chooses its actions by following a policy π, which yields a probability $\pi(s, a) = p(a \mid s)$ of taking a particular action from the given state. An optimal policy π^* maximises the value function

$$V^*(s) = \max_a [R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') V^*(s')], \tag{1}$$

where $\gamma \in [0, 1]$ discounts future rewards. In essence, a model following an optimal policy selects the action that in the current state of the environment maximises the sum of the immediate and discounted future rewards, assuming future compliance with optimal policy. In situations of partial observability, the current state of the environment is not known but approximated from the history of observations. In those cases, the agent maintains a belief state, which is updated by considering the previous prior belief state b, an observation, and the posterior belief state b' via Bayesian belief updating:

$$b(s') \propto O(s', a, o) \sum_{s \in S} \hat{T}(s, a, s') b(s), \tag{2}$$

where $\hat{T}$ is a learned world model that approximates the true transition model T.

Our hierarchical model employs four distinct but interrelated agents: vision, supervisory control, pointing, and proofreading. The first two are described as MDPs, and pointing and proofreading as POMDPs. Each agent or subtask model μ has its own value function V^*_μ and, therefore, its own policy π^*_μ. Figure 2 illustrates the full model architecture. The supervisor observes and guides the subtask agents. The pointing agent's task is to move a finger to the requested key, balancing between speed and accuracy. Both the target key and the desired speed–accuracy trade-off are reported to the pointing agent by the supervisor, causing the finger to move over the typing device's keyboard. Because this pointing action contains noise and since the agent has uncertainty about the correct action, sometimes the typed text contains errors. To address this, the proofreading agent follows what is being typed and stands ready to detect errors. It can do so either by observing the pointing agent's actions and predicting their likelihood of resulting in a mistake or by directly executing a proofreading action, which yields more direct and certain information about possible errors.

The demands of pointing and proofreading highlight the key constraint in touchscreen text entry. When physical keycaps are absent, the pointing agent is uncertain about the current finger position. The uncertainty can be reduced by visually sampling the area where the finger currently is located. To this end, the supervisor can request the vision agent to bring the gaze close to the finger. However, the same gaze is needed by the proofreader for establishing certainty about the presence of typing errors. It is therefore the task of the supervisor to allocate the limited visual attention between the competing subtask agents, and to instruct these agents for efficient typing. The source code for our model is available at https://github.com/aditya02acharya/TypingAgent.

3.2 Pointing

The pointing agent μ_p is responsible for simulating physical interaction with the touchscreen device. Its state $s \in S_p$ is the current location of the finger (represented as one-hot encoded vector of size 44) on the device, with the state space S_p being the set of all unique interactive elements (i.e., keys) of the device. In addition, there are state indicators or 'slots' for the requested target (represented as one-hot encoded vector of size 32) and the desired speed–accuracy trade-off. The action space A_p contains all device elements that can be interacted with, such that there are, in total, 10 actions for each element: 'move' and 'move and peck' in five separate speed–accuracy trade-off configurations. In addition, there is a general 'no-op' action, doing nothing. The five 'move' actions (separate for each device element) take the finger from its current point to the target element either as quickly or as accurately as possible, or in some balance between these two extremes. The five 'move and peck' actions take the finger to the target in a similar fashion, then press it.

The transition function T_p provides the probability $T_p(s, a, s') = p(s' \mid s, a)$ of moving to a new location, given the previous finger location and the action. The motor noise depends on the finger's accuracy and on the distance travelled, as predicted by the WHo pointing model [25, 26], described in detail in Supplementary A.5. It expresses pointing ability as a weighted homographic curve with speed and accuracy extremes. Any pointing action can be located along the curve, which describes it in terms of movement time (speed) and relative endpoint spread (accuracy). When adding noise to the finger endpoint, we nevertheless constrain the movement to the limits of the physical device, as this is a realistic assumption with humans.

The observation function $O_p(s_v, s_p, o) = p(o \mid s_v, s_p)$ gives a probability of making an observation $o \in \Omega_p$, given $s_v \in S_v$ (the current state of vision) and the state of the finger $s_p \in S_p$. The intuition is that the pointing agent, since it can locate the finger

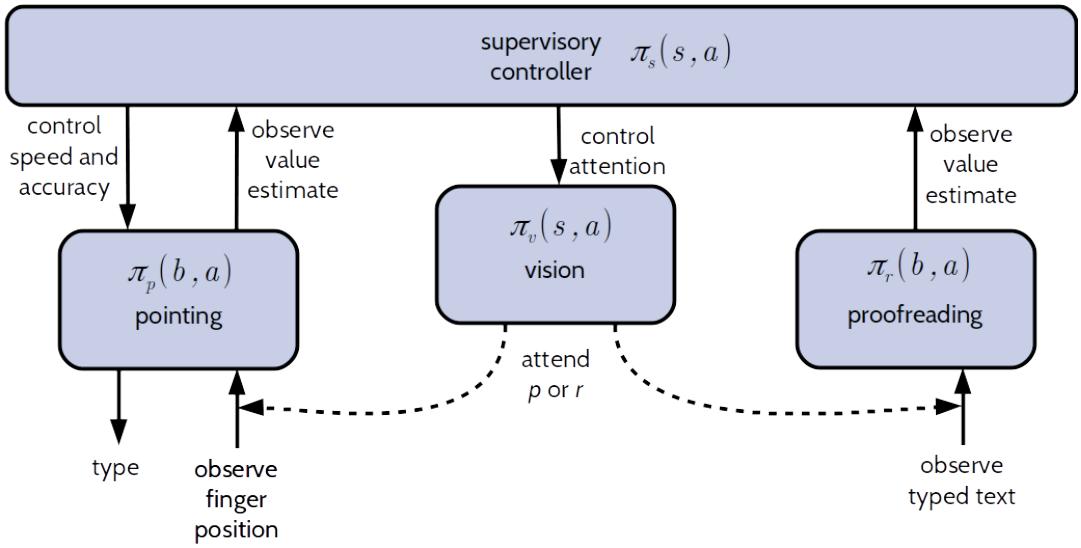


Figure 2: An overview of our model of touchscreen typing as optimal supervisory control. It is composed of four distinct agents: supervisor, pointing, vision, and proofreading. The three subtask agents interact with the task environment (the mobile device): the pointing agent may touch the keyboard's keys and make observations about current finger position. The proofreading agent observes the typed text and ascertains the presence of errors. To model limited visual acuity, vision is shared between pointing and proofreading, restricting these agents' ability to simultaneously observe the environment. Each agent is in a state s or holds a belief b about that state, and it applies a policy π to perform an action a. The policy also predicts the value of this action. The supervisor decides how vision is allocated between these subtasks and dictates the speed–accuracy trade-off for the pointing agent, in light of its observations of current subtask values.

visually, is able to observe the finger's position. The pointing agent holds a belief state b_p, which is a probability distribution over S_p; that is, each possible position of the finger has an associated probability. Bayesian belief updating (Equation 2) is used to compute the posterior belief state, for a given previous belief state, observation, and learned transition model that approximates the finger movement model. When vision guides the finger, this belief accurately corresponds to the finger's true position. However, when the gaze is elsewhere, the pointing agent updates its belief in line with the stochastic transition model, thus producing increasingly less certain beliefs about the current finger position.

The pointing agent is rewarded with $r = \sigma_d \cdot h - (1-\sigma_d) \cdot mt$, where $h = 1$ if the finger model presses the requested target and $h = 0$ otherwise; $\sigma_d \in [0, 1]$ is the requested speed–accuracy trade-off (parameter values near 1 prioritise accuracy over speed); and mt is the finger movement time, computed via the above-mentioned WHo model. The intuition behind the reward function is that the model is rewarded for hitting the requested target but the reward's amount depends on the requested weight to give to finger accuracy vs. speed. As movement time is always discounted from the reward, higher σ_d values result in behaviour wherein the finger model emphasises speed at the cost of hitting the target, and lower values favour hitting the target, with less emphasis on minimising movement time. The optimal policy for moving finger to type by maximising the reward value is found using the *Deep Q-network* (DQN) [40] algorithm.

3.3 Vision

The vision agent μ_v is responsible for lowering the uncertainty of the subtasks by granting them the ability to make accurate observations about states of the environment. It simulates eye movements over the touchscreen device, providing visual attention to guide finger movements and proofread the text entered. We utilise the eye-movement model EMMA [51], a sequential description of the process of eye movements and visual attention. It describes how requests to encode a certain visual element result in shifting of attention with potential eye movements and the subsequent encoding of the target in a certain time span predicted by the model. Supplementary A.6 describes the model in detail.

Vision is specified as an MDP, where $s \in S_v$ contains the visual world as all the individual elements visible, the current fixation location, any visual element that is currently encoded, and the requested visual target. An action $a \in A_v$ creates a visual encoding of a single element of the visual world, with each individual element having its own dedicated action. The transition function T_v provides the probability $T_v(s, a, s') = p(s' \mid s, a)$ of moving to a new state. Given an action to visually encode an element, the transition function changes the state of the currently encoded visual element to correspond to the new one and optionally changes the fixation location (as specified by the EMMA model). After its action to encode a new visual element, the agent is given a reward $r = f - mt$, where $f = 1$ if the visual element of the current state encoded is the visual element requested and $f = 0$ otherwise. The eye-movement and encoding time is mt, determined by the EMMA model. The

optimal policy for moving eyes to target key is found using the *Q-Learning* algorithm [58].

The vision agent described thus far is fully observable in that the state is completely known to the model. This corresponds to a situation wherein the location of each visual element is known to the agent, as in the case of typists who are experienced with their keyboard layout. Our model can be extended to partially observable instances – e.g., to simulate behaviour with previously unknown or partially learned layouts [34], or layouts with dynamically changing content.

3.4 Proofreading

The proofreading agent μ_r is tasked with detecting errors in the text typed thus far by the model. It does this by maintaining and continuously updating a prediction value for the presence of errors in the typed text. The main task of the proofreader is to hold a belief about an error being present in the text stream and conduct proofreading when the probability of error is deemed high enough to warrant it. To this end, the state $s \in S_r$ simply indicates whether there is an error (one or more) in the text just typed, but the proofreader cannot directly observe this state. The agent can perform an action $a \in A_r$ to make a proofreading pass over the typed text or do nothing (a 'no-op' action). The proofreader is connected to the pointing agent: the transition function T_p for the proofreading task simply switches from no-error to error state when the keystroke just made introduces an error into the text stream. No-error state is restored only after the incorrect keystroke has been eliminated by means of error correction, such as backspacing.

After each action $a \in A_p$ by the pointing agent, the proofreading model makes an observation $o \in \Omega_p$, which indicates whether the keypress resulted in an error or not. The observation function $O_p(s', a_p, o) = p(o \mid s', a_p)$ gives the probability of having noticed that an error was made, given the finger movement. This reflects the possibility of the typist spotting a typo immediately (for instance, thanks to seeing the finger touch the wrong key). The observation is used to update the belief state b_r, via Equation 2. In addition to direct observation, the proofreading agent tries to predict the occurrence of errors from the finger movements.As this direct observation is noisy and since the eye is not necessarily on the finger, the agent only holds a certain belief about the presence of a typing error. The probability starts at 0 when a new transcription task begins and right after a proofreading action has been completed with no errors detected. From there, it starts to grow as the likelihood of error increases.

Upon each action (proofreading or no-op), the agent receives a reward $r = f - mt$, where $f = 1$ if the proofreading action uncovered an error and 0 otherwise, and mt is the time consumed by the proofreading, computed via the EMMA eye-movement model by assuming that one word of entered text requires one fixation and encoding by the model (see Supplementary 1.a for details). The no-op action takes no time, so the reward after this action is always 0. The agent learns the optimal state-action mapping using Q-Learning. The agent's behaviour as dictated by these rules results in favouring proofreading when the probability of error is high and giving proofing attempts lower priority when the agent's belief indicates low likelihood of error. Ultimately, the supervisor is responsible for deciding when the value is high enough for moving the vision to aid in proofreading.

3.5 The Supervisor

The role of the supervisor is to manage the three subtask agents (pointing, vision, and proofreading) such that they produce coherent and efficient transcription typing behaviour with touchscreen devices. The supervisor's state $s \in S_s$ describes the current $V^*_\mu(s_\mu)$ of subtask agents μ_p and μ_r. This means that the supervisor tracks the pointing and proofreading agents' current values and uses the information to make decisions about what to do next. In addition, the supervisor knows the target sentence and has a pointer denoting how much of that sentence has been typed at any given time. The supervisor takes an action $a \in A_s$ to adjust the slots of the subtasks. For instance, the supervisor selects the σ_d value of the pointing agent, thus determining the desired speed–accuracy trade-off of the finger. Because the pointing agent has been trained with different σ_d values, it has learned the correct actions corresponding to this instruction, given where it is asked to point to next. The supervisor also instructs the vision and the pointing agent on their next target. Thus, it is up to the supervisor to decide when the vision is needed by the proofreading agent and when the gaze should be on the keyboard to guide finger movements. The supervisor is rewarded only at the end of a typed sentence, with $r = -mt - e$, where mt is the total time spent typing and e is the amount of error in the sentence, computed as Levenshtein distance. The agent learns to optimally control how to deploy the limited human vision using a type of *policy gradient method* called 'PPO' [55].

The intuitive explanation for why the supervisor observes the value estimates from the subtasks is that this lets it decide which task – pointing or proofreading – currently needs the vision more. The values for the pointing agent's actions are high when it is confident about the finger's position and low when there is considerable uncertainty. In the latter situation, the pointing agent has learned to wait, moving the finger close to the next target, but not pecking (due to this being probably an error), and then taking the no-op action, which brings no movement-time penalty. At the same time, the value given to proofreading is high when the proofreading agent expects to spot an error. Observing these two values lets the supervisor learn a control policy of proofreading only if an error is likely enough to be present and otherwise using the vision to guide pointing. However, this policy is conditional to the actual value estimates of the subtask agents. For instance, in a scenario wherein the finger does not need visual guidance, a likely scenario with an expert typist or physical keycaps, the supervisory control policy reflects this by keeping the attention mainly on proofreading.

3.6 Extending the Model for Two-Thumb Typing

The model described above simulates typing with one finger, but it can be extended into two-thumb typing, as we also demonstrate further on in this paper. Two-thumb typing is both a generally faster and a more popular way to enter text on touchscreens [48]. This improvement in typing efficiency is due to multiple factors, such as lower IKIs, arising from finger alternation and shorter travel distances, and the ability to focus more on the text-entry area since

the finger movements are in a more restricted area [31]. Interestingly, the finger-to-key mapping in two-thumb typing is not always constant; in some circumstances, certain keys toward the centre of the keyboard may be pressed by either thumb [31]. This is probably due to a learned control policy whereby typists opportunistically choose the finger closest to the key, physical limitations (such as finger length and potential collisions between fingers) permitting. Our choice of modelling paradigm is especially suitable for simulating such dynamic choice of finger-to-key mapping. Though a complete two-thumb model is left for future work, we will demonstrate the capacity for it by creating a model that uses two pointing agents instead of one, with keys mapped to them such that the fingers do not cross paths at any point. We foresee the model's architecture being augmented with rules governing the parallel motion of two fingers, also considering possible collisions, and an additional subcontroller, orchestrating finger movements and avoiding such collisions, for a more realistic simulation.

4 EVALUATION

To evaluate our model, we compared its simulation results to data from human transcription typing on a touchscreen device. All the model's parameters were taken from the literature or were task-based, with an attempt to keep the tasks as similar as possible to the humans'. Therefore, the correspondence of its predictions with the human data stems from *realistic model specification*. No human data were necessary for training or calibrating the model. This is an important feature of our modelling paradigm: we look for the behaviour that emerges from objectively established constraints, resources, and goals rather than predict performance through mathematical formulae that hide these factors in parameters that are empirically tuned whenever circumstances change.

4.1 Stimuli and Tasks

The human data for our evaluation come from a study [31] that generated detailed finger- and eye-tracking data from 30 participants transcribing 20 sentences each, sampled from a 75-sentence corpus (data freely available via https://userinterfaces.aalto.fi/how-we-type-mobile/). The work examined two task conditions, one-finger and two-thumb typing. While our main focus was on how well our model's predictions matched the one-finger dataset, we tested the initial two-finger model with human data from the two-thumb typing condition also.

Our simulation replicates the task environment of the study with humans. The device used by our simulation matches the design in the original study: the keys are in the same positions and of the same dimensions (supplement B provides an image), and the device (a Samsung Galaxy S6, 1440 × 2560, 577 pp, 5.1″ screen) presents the text similarly in the upper part of the view. Also, the sentences used in testing of the model are the ones the human participants typed. To avoid overfitting to a specific test corpus, we trained the model with a set of sentences different from those used with the humans (Supplementary C describes the data). All subtask models were trained until convergence; then, the same was done for the supervisor. After convergence, the simulation ran through each sentence in the test set, in 30 independent runs (but using the same trained model), to produce variations in how the model typed individual sentences.

4.2 Data Analysis

4.2.1 Details of finger and eye movements. For a 'sanity check' of the model's behaviour, seeing it 'in action' is important. We can compare a typical simulation of typing a sentence to how the equivalent task looks when a human performs it. For this, we employ two ways of illustrating finger and eye movements, both adapted from the techniques in the original study: 1) creating a heatmap of these movements for one sentence, thus visualising where they occur, and 2) illustrating key-by-key finger and eye distances during typing of a sentence, to demonstrate how eye movements guide the finger, how the two sometimes move in parallel, and how proofreading moves the eyes away from the keyboard and briefly suspends typing.

4.2.2 Aggregate metrics. In more quantitatively oriented testing, we can evaluate how well our predictions match the human data for selected typing metrics, by 1) analysing whether the prediction for a given metric lies within the range observed with humans and 2) examining how far the model is from the humans' grand-mean value for a given metric, both in absolute mean differences and in relative terms via the standard deviation from the human data. In a sense, we are testing whether the model's predictions are plausible, human-like ones or unrealistic outliers. We require the predictions to be within one SD, as most humans (about 68%) should be within this band around the average value [28]. Furthermore, we consider a result to be feasible, though possibly an outlier, if within a 3 SD distance of the mean (this range should cover about 99.7% of humans). Values beyond this would be considered unrealistic predictions by our model. We chose the following metrics for this testing, on the basis of their use in previous studies of typing, as reviewed in Section 2.

- *Inter-key interval* [21]: the time between two consecutive keypresses.
- *Words per minute* [21]: the number of 'standard words' (five characters of the text ultimately entered) divided by total time spent.
- *Chunk length* [31]: the number of characters per 'chunk', with a an IKI greater than the sentence-average IKI indicating a chunk boundary.
- *Backspaces* [48]: the number of Backspace presses during typing.
- *Immediate backspacing* [3]: the frequency of the user noticing an error immediately, then pressing Backspace.
- *Delayed backspacing* [3]: how often the user notices an error only later, after having typed further characters and thereby needing multiple backspaces to correct the text.
- *Fixation count* [31]: the total number of eye fixations, separated by saccadic movements.
- *Gaze shifts* [31]: the number of glances into the text area from the keyboard, indicating proofreading or monitoring of error correction.
- *Gaze keyboard time ratio* [21]: the percentage of the time that the gaze is on the keyboard.

- *Finger travel distance* [31]: the sum-total distance travelled by the finger(s).

4.2.3 Typing dynamics. As discussed in Section 2, certain stable patterns can be seen in touchscreen typing. For example, speed (in WPM) exhibits a strong correlation with the amount of backspacing during typing. We can extend the example: a larger number of Backspace presses is generally associated with more frequent proofreading, which correlates, in turn, with overall typing speed since typing on a touchscreen without one's eyes on the keyboard is difficult. Careful multivariate analysis is required to unconfound such associations and reveal which metrics truly contribute to typing performance, in what ways. Evaluating our simulation model entails the same analyses for the simulated and the observed data, along with comparison of the direction and magnitude of the β coefficients in the multivariate statistical models. We can consider the predictions to match human results well when the difference between the βs does not exceed 0.2 (i.e., a small effect [18]), and acceptable as long not exceeding 0.5 (a medium effect). Because of the nested structure of the data (that is, individual sentences being typed by multiple subjects, with individual participants contributing multiple sentences to the dataset), we utilised multilevel regression for this analysis (`lme4` library in R).

4.3 Results

4.3.1 Details of finger and eye movements. Heatmap comparison between the data from humans and the model is shown in Figure 1. In both bodies of data, it is clearly apparent that the focus is most often on the keyboard. The eyes guide the finger movements, and very few of the fixations land in the text-entry area for proofreading. The greatest distinction between the heatmap for the humans and that for the model is that the former shows a larger footprint, indicating either larger movement arcs or noisier sensory readings.

Figure 3 shows a given sentence typed by both the model and a human participant. Most importantly for the model's realism, similar eye–hand movement patterns are evident. One can see how the eyes guide the fingers by moving toward the same target. Programming the eye movement takes some time for the human visual system [51], manifested as a flat line before the fast saccadic movement. While many models simulating touchscreen typing require the eye to move to the target key before finger movement occurs [13, 32, 54], ours demonstrates parallelising finger and eye movements similarly to humans. The finger can start moving toward the target before the eyes have reached it: they will still arrive in time to guide the finger's final motion. At times, when foveated vision already covers the target key, no eye movements are necessary (e.g., when 'o', 'p', and 'i' are typed in sequence).

4.3.2 Aggregate metrics. The predicted and observed mean values for the selected metrics are listed in Table 2. Most of the values fall within realistic ranges for human production. One can see that the WPM values, on the whole, are near the mean value from the human data, and the model-predicted IKIs are likewise average in human terms. As for error-correction, the model clearly attempts to avoid errors, so it predicts small amounts of immediate and delayed backspacing alike; however, both are within plausible human ranges. Varying the parameters of the finger model could yield different WPM or IKI values, as well as different amounts of backspacing, and resulting adaptive changes in gaze deployment [53, 54].

The model accurately predicts the fixation count for the typing of a sentence. Proofreading behaviour, in terms of how often the model shifts gaze to the text-entry area, is consistent with what has been observed in humans. Although the gaze in the model spends a larger proportion of the total typing time on the keyboard than humans generally do, the value is still realistic. The eye-movement model might underestimate the time to proofread words, because it was originally parameterised from reading rather than proofreading tasks [51].

4.3.3 Typing dynamics. Reporting on our investigation of how well the model produces realistic dynamics of touchscreen typing, Table 3 shows linear regressions with either WPM or gaze shifts as the dependent variable. We wish to emphasise that the analyses, while using linear models, are neither attempts at statistical prediction nor statements pertaining to a causal connection. Rather, they let us investigate the various ways in which these metrics are mutually related. The small overall differences between the model's predictions and the observations from humans show that our model can simulate the dynamics well. For instance, non-surprisingly but importantly for realism, the number of Backspace presses per sentence has a large negative correlation with WPM values: making and correcting errors slows the user's typing considerably. Having to correct errors shows a positive correlation with gaze shifts, due to the need to proofread and monitor the corrections. Likewise, there is a high negative correlation between WPM values and gaze shifts. To unconfound these correlations, we add the counts for both Backspaces and gaze shifts to the regression. A smaller but still negative effect of gaze-shifting then becomes apparent. This was noted in the report on the original study: even when one statistically controls for the number of corrections made, proofreading takes time so is negatively correlated with WPM.

These results highlight the difficult optimisation problem facing touchscreen typists: they should minimise proofreading time yet detect and correct errors as quickly as possible. Additional analysis found support for this conclusion in both datasets. Errors that are swiftly detected and corrected have a smaller WPM impact than delayed backspacing. Finally, a correlation is clearly visible between proofreading activity, measured in terms of gaze shifts, and remaining errors, judged via the corrected-error rate. While the model's βs differ from those of the original data by more than 0.2 in three places, even these show the same direction of the effect, without a large deviation in magnitude.

4.4 Evaluating Two-thumb Typing

Our work-in-progress two-thumb typing model permits considering the impact of an additional finger on certain typing metrics. Comparison with human data in Figure 4 shows that the model replicates some, though not all, of the effects. As the model predicts, typing speed rises, since the fingers do not have as far to travel. Also, the IKIs fall for both humans (384 ms to 272 ms) and the model (399 ms to 376 ms), although, possibly because of its strict finger-to-key mapping, the model benefits less than humans do in this regard. Further examination uncovers a curious adaptation in humans, not predicted by the model. As the original report indicates [31],

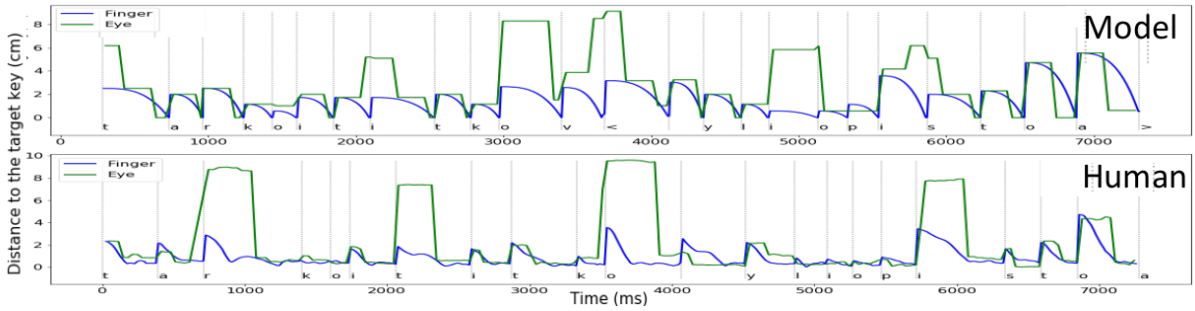


Figure 3: Key-by-key distances of the eyes and the finger from the target, for the model and one human participant (subject 117 in the data), typing the same sentence. Note the slight difference between the scales. Clearly visible are eye movements to the text-entry area for proofreading. < denotes pressing Backspace, and an empty slot represents the spacebar. For the model's data, the finger's trajectory from one position to another is quadratically interpolated.

Table 2: Grand means for selected typing-performance metrics: aggregate values for the human subjects and the simulations, the absolute difference, and difference in the SD of the variable in the human data. We consider a prediction good (green shading) if it falls within the range of the human data and is within one SD of the human mean and we deem the values still acceptable (realistic but outliers relative to the humans, in orange) if within three SDs from the human mean.

Metric	Human mean	Min. human value	Max. human value	Human *SD*	Model mean	Mean diff.	Diff. in *SD*
IKI (ms)	380.94	311.35	514.25	50.95	398.85	17.92	0.35
WPM	27.19	19.12	33.30	3.61	25.22	1.97	0.55
Chunk length	3.98	3.44	5.13	0.41	3.90	0.08	0.20
Backspaces	2.61	0.35	8.80	1.81	1.49	1.13	0.62
Immediate backspacing	0.40	0.00	1.05	0.26	0.31	0.09	0.35
Delayed backspacing	0.63	0.10	2.15	0.47	0.47	0.17	0.35
Fixation count	24.04	17.75	36.38	4.56	23.21	0.83	0.18
Gaze shifts	3.91	1.19	8.69	1.50	4.16	0.25	0.17
Gaze keyboard time ratio	0.70	0.36	0.87	0.14	0.87	0.18	1.31
Finger travel distance (cm)	25.29	20.81	27.64	1.33	22.09	3.20	2.41

humans make more errors (and, hence, perform more backspacing) when typing with two thumbs, but there are fewer gaze shifts than one might expect. The model replicates the reduction in gaze shifts; however, it produces fewer errors, owing to the smaller finger movements and the resultant decrease in noise. We hypothesise that the reason our model fails to reproduce this pattern is that we did not constrain the physical movements of the two thumbs. Human participants can move their fingers rapidly without losing track of their position due the movement to being constrained to the corners of the physical device [46]. While the number of errors increases slightly, overall performance nonetheless improves.

5 APPLYING THE MODEL: AN INTELLIGENT TEXT ENTRY AID

Our model, thanks to being rooted in the approach of optimal supervisory control, is able to simulate the human ability to adapt quickly, finding efficient control strategies as circumstances and therefore the bounds of the task change. In the domain of touchscreen text entry, it appears that most typists use some sort of intelligent text-entry aid [48]. A major roadblock has stood in the way of these aids' development, at least until now: computationally evaluating them is very difficult, because the models used for touchscreen typing do not adequately predict the adaptive changes implied by such aids. For instance, in typing with intelligent error-correction or a good word-prediction system, more errors are permitted, so users are free to type more quickly, potentially with lower proofreading requirements. Yet existing models cannot simulate such changes well enough – they either neglect to predict errors or do not encompass error-correction as the dynamic and complex phenomenon that it is known to be.

By way of demonstrating our model's capability of simulating adaptation to intelligent text-entry aids, we designed an experiment wherein the model was exposed to three distinct text-entry designs: 1) the standard keyboard described in the validation section, above; 2) inclusion of intelligent error-correction that has a 50% probability of correcting a mistyped character to the right one, as long as the incorrect key is adjacent to the correct one; and 3) a design that increases the latter probability to 90%. Both of the intelligent designs

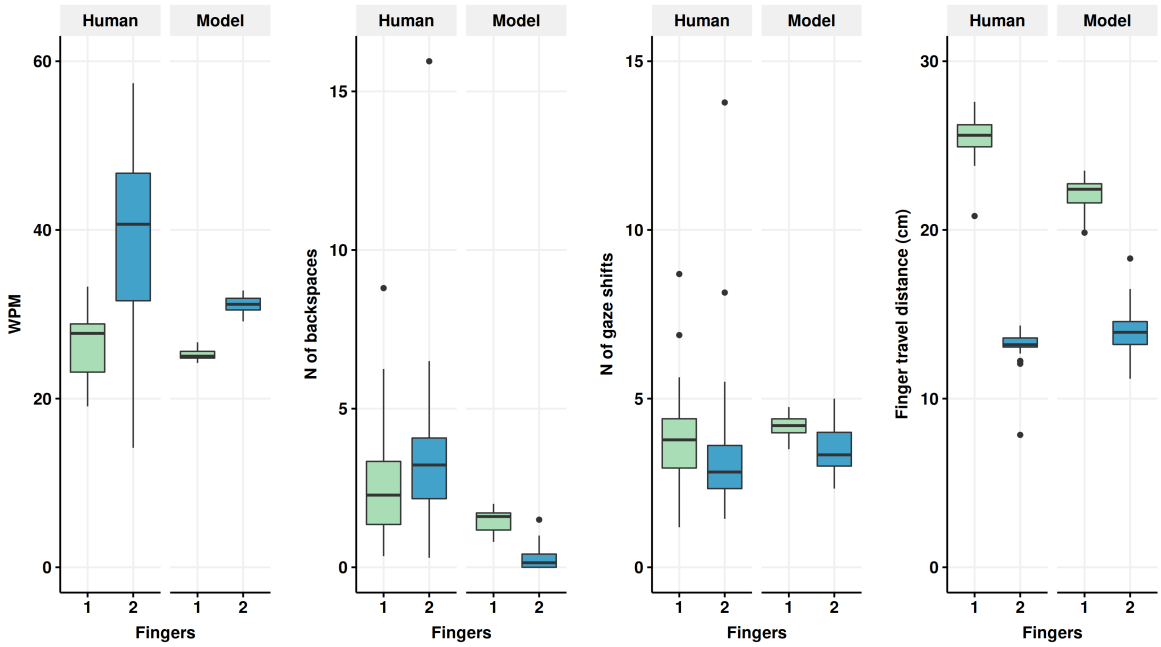


Figure 4: Comparison of selected metrics between one- and two-finger typing, for human and model data.

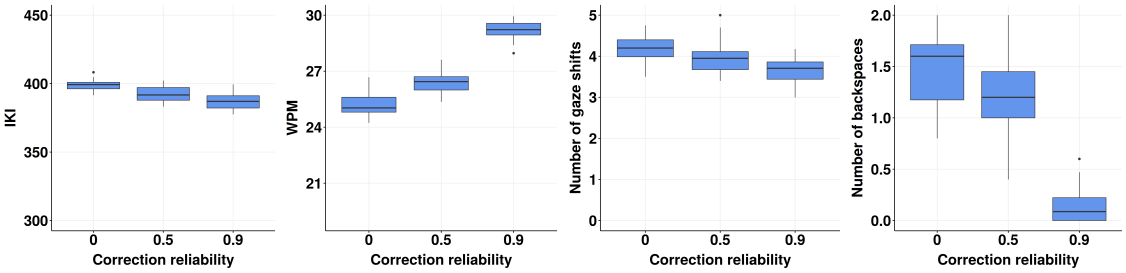
Table 3: β coefficients for linear regressions, for both human and model data, correlating either typing speed (WPM) or proofreading frequency (gaze shifts) with various other metrics ('BS' = backspaces). We consider the match good (green shading) if the difference between the two values does not exceed 0.2 and acceptable (yellow shading) if greater but not exceeding 0.5.

Response variable	Effect	Human β	Model β	β diff.
WPM	N of BS	−0.75	−0.81	0.06
WPM	N gaze shifts	−0.63	−0.43	0.20
WPM	N of BS	−0.59	−0.78	0.19
	N of gaze shifts	−0.27	−0.11	0.16
WPM	N of immediate BS	−0.32	−0.50	0.18
	N of delayed BS	−0.67	−0.88	0.21
WPM	N of immediate BS	−0.24	−0.48	0.24
	N of delayed BS	−0.54	−0.85	0.31
	N of gaze shifts	−0.22	−0.07	0.15
Gaze shifts	N of BS	0.52	0.41	0.11
Gaze shifts	N of immediate BS	0.29	0.36	0.07
	N of delayed BS	0.48	0.45	0.03

are somewhat unrealistic, as we merely exploited the fact that the simulation knows the sentence that is currently being transcribed, which permits us to inject the correct character with the desired probability. The goal with this simplified design, however, is to demonstrate our model's ability to adjust to intelligent text-entry aids, not to present an actual design for such assistance.

The results of the experiment are illustrated in Figure 5. Obvious changes, which could be predicted with non-adaptive models as well, appear in the higher typing speed, in WPM, and the reduced need for error-correction. Speaking to the model's ability to adapt to the task constraints, IKIs are shorter for conditions of more reliable automatic error-correction. This is understandable: as the model learns that even faster pointing movements still yield a correctly typed word, the supervisor agent can instruct the finger agent to trade accuracy for more speed [49]. As long as the faster but less accurate keypresses are confined to neighbouring keys, meaning that the errors are likely to get corrected, there is less need for proofreading, measured in the number of gaze shifts to the text-entry area. However, as the figure shows, proofreading remains necessary even with highly reliable intelligent error-correction (Figure 1b shows a scanpath for typing a sentence in the 90% condition). This is because the model may still overshoot a key that is at the edge of the keyboard, thereby pressing no key at all, and the intelligent aid responds only to actual typos (i.e., insertion errors or transposition, not deletion errors). Likewise, if the finger overshoots further than to the key adjacent to the intended one, no correction is done. Errors of both sorts, remaining uncorrected by the intelligent aid, quickly compound as the model enters more characters. Therefore, the model still needs to keep track of what actually has appeared in the text-entry area, for knowing which keys to target next.

The brief experiment with intelligent text-entry aids reported on here demonstrates the power of the theory of optimal supervisory control. In line with the assumption of optimal adaptation



(a) Inter-key interval **(b) Words per minute** **(c) Number of gaze shifts** **(d) Number of Backspace presses**

Figure 5: Four performance metrics for the three devices used: a standard keyboard and two with intelligent error correction, of differing reliability. Note that the lower bounds for y-axes for IKI and WPM have been set to human minimum values from [31] raher than 0.

to task constraints, strategic adaptations such as those reported upon here can be said to emerge from how the task and the agent are specified. There is no need for heuristic assumptions about proofreading frequency or the correct speed–accuracy trade-off (as exemplified in the literature [54]), because these strategies are discovered during learning of the optimal supervisory policy. Importantly, the designer/modeller need not make any assumptions as to how the users are going to utilise the design – this policy is discovered entirely under the assumption of optimal adaptation. We therefore foresee multiple avenues for advancing the design of intelligent text-entry aids via our model by experimenting with new ideas in the manner presented here.

6 DISCUSSION AND CONCLUSIONS

Humans have an astonishing capacity to adapt to changes in the task environment. A perfect demonstration of this is how we can adopt new technologies quickly, even in an age involving their constant change. Entering text via touchscreens has been a popular method of communication for merely a decade or so, yet many users have adapted to using them – at impressive speeds. Furthermore, both the availability of various intelligent text-entry methods and their obvious popularity [48] make it clear that humans do not just translate their existing skills across domains; we are able to learn completely new paradigms of interaction. Hence, HCI researchers face the challenge of identifying a unifying theory and mechanism that explains our touchscreen typing abilities. With the work reported upon in this paper, we have tackled that challenge, demonstrating that the theory of optimal supervisory control is able to explain and model the plethora of ways that humans interact with touchscreen keyboards in transcription typing.

Our model assumes a hierarchical architecture of subtask and supervisory control. The controllers are implemented as agents encountering the problem of choosing from among competing actions to reach a given goal. These agents are *adaptive*, learning to select effective actions after accumulating experience. Through this assumption in our model, we have achieved credible simulation of the orchestration of the eyes and fingers moving over the touchscreen device. The resulting replication of how humans type serves as a reminder to researchers and designers who work with touchscreen text entry: typing is not purely a matter of motor performance. The lens of optimal supervisory control offers an exciting new account of visuo-motor adaptation that can predict the emergence of eye–hand movement strategies in typing. We can more powerfully model realistic human data in typing without manually coding how the eyes and fingers move – we can let those policies emerge, just as they do in real life.

With the parameters being obtained from the literature or determined in light of the task, the simulation's ability to replicate human-like typing patterns and aggregate metrics is due to specification of the model, not tuning of parameters against target data. Nevertheless, we also foresee an important avenue for future research with regard to parameter inference and the subsequent simulation of *individual-specific differences*. Touchscreen typists are known to manifest various eye–hand co-ordination strategies [31], in part due to differences in abilities, connected with such factors as finger accuracy and experience with the given layout [54]. For instance, simulation of how knowledge of the layout elements' positions and features affects typing, especially with regard to eye-movement patterns [34, 35], fits well within our framework. Identifying those parameters of our model that best capture such differences and designing a corresponding parameter-inference scheme (in the mode of prior literature [37]) would allow predicting individual- and occasion-specific aspects of adaptation to interfaces. In an even more advanced line of UI development, it might be possible to infer the parameters for a user 'online' (that is, during use) from the user's behaviour. Thus parameterised case-specifically, our model could then be used to automatically test various 'what if' designs, thereby enabling continuous UI optimisation. Such optimisation is rendered especially important by the great variety in several abilities: expertise (with typing and in particular conditions), finger accuracy, visual acuity, etc. [54]. Such optimisation is rendered especially important by the variability in abilities: expertise, finger accuracy, visual acuity, etc. [54].

The development of the model as presented here was limited to a setting of transcribing text from a pre-selected corpus while seated and not experiencing any distractions. In real life, typing is rarely this straightforward. Users generally have to compose the message in their mind while typing, they might be walking

while they type with the touchscreen device, and even the device itself may present distractions (e.g., social-media notifications) that result in dividing one's attention. While none of these factors fell within the scope of our model, we would expect it to be possible to account for them within our general architecture. One could conceptualise the activities listed, such as composing messages or reading notifications, in terms of particular task models each set within a hierarchy of varying complexity, with these then handled in parallel with the typing model through a 'superhierarchy' of concurrent tasks. Multitasking of this nature would influence the typing conditions in several ways (for instance, any of several tasks might require vision resources); however, our model is already able to adapt to such changes in the task environment.

In commitment to advancing open science and facilitating adaptation further, we publish all our model code in open-access form. Furthermore, to assist with applying the model as a design tool, we have implemented a UI for easily running simulations with it. The UI, described in Supplementary D, allows manipulation of the model's basic parameters, along with the design of the typing device. The effects of particular changes in settings can be investigated within the UI or exported. Formally defined models for well-grounded simulation of adaptive user behaviour are a key instrument in design, especially when the design spaces are large and there are numerous possible ways in which a user may adapt to design changes. Such models must be rooted in psychologically valid theories of human behaviour and be supported empirically. We have paid great attention to these underpinnings. Finally, while the focus here has been on touchscreen text entry, our theoretical approach is not limited to this domain. In fact, many computerised tasks that require human control can be understood in terms of optimal-control policies and supervisory resource allocation.

ACKNOWLEDGMENTS

This research has been supported by the Academy of Finland projects BAD and Human Automata, and Finnish Center for AI (FCAI). Anqi Yang helped with the UI development. The project has a website at https://userinterfaces.aalto.fi/touchscreen-typing/

REFERENCES

[1] Aditya Acharya, Andrew Howes, Chris Baber, and Tom Marshall. 2018. Automation reliability and decision strategy: A sequential decision making model for automation interaction. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 62. SAGE Publications, Los Angeles, CA, 144–148. https://doi.org/10.1177/1541931218621033

[2] John R. Anderson, Daniel Bothell, Michael D. Byrne, Scott Douglass, Christian Lebiere, and Yulin Qin. 2004. An integrated theory of the mind. *Psychological Review* 111, 4 (2004), 1036–1060.

[3] Ahmed Sabbir Arif, Sunjun Kim, Wolfgang Stuerzlinger, Geehyuk Lee, and Ali Mazalek. 2016. Evaluation of a smart-restorable backspace technique to facilitate text entry error correction. In *CHI '16: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, CA). ACM, New York, NY, 5151–5162. https://doi.org/10.1145/2858036.2858407

[4] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2009. Analysis of text entry performance metrics. In *2009 IEEE Toronto International Conference Science and Technology for Humanity (TIC-STH)*. IEEE, New York, NY, 100–105. https://doi.org/10.1109/TIC-STH.2009.5444533

[5] Ahmed Sabbir Arif and Wolfgang Stuerzlinger. 2010. Predicting the cost of error correction in character-based text entry technologies. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems – CHI 2010*. ACM, New York, NY, 5–14.

[6] Patrick Armstrong and Brett Wilkinson. 2016. Text Entry of Physical and Virtual Keyboards on Tablets and the User Perception. In *Proceedings of the 28th Australian Conference on Computer-Human Interaction* (Launceston) *(OzCHI '16)*. ACM, New York, NY, 401–405. https://doi.org/10.1145/3010915.3011006

[7] Shiri Azenkot and Shumin Zhai. 2012. Touch Behavior with Different Postures on Soft Smartphone Keyboards. In *Proceedings of the 14th International Conference on Human-computer Interaction with Mobile Devices and Services* (San Francisco, CA) *(MobileHCI '12)*. ACM, New York, NY, 251–260. https://doi.org/10.1145/2371574.2371612

[8] Nikola Banovic, Varun Rao, Abinaya Saravanan, Anind K. Dey, and Jennifer Mankoff. 2017. Quantifying aversion to costly typing errors in expert mobile text entry. In *CHI '17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 4229–4241. https://doi.org/10.1145/3025453.3025695

[9] Nikola Banovic, Ticha Sethapakdi, Yasasvi Hari, Anind K. Dey, and Jennifer Mankoff. 2019. The limits of expert text entry speed on mobile keyboards with autocorrect. In *Proceedings of the 21st International Conference on Human–Computer Interaction with Mobile Devices and Services*. ACM, New York, NY, 12.

[10] Xiaojun Bi, Yang Li, and Shumin Zhai. 2013. FFitts law: Modeling finger touch with Fitts' law. In *CHI '13: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris). ACM, New York, NY, 1363–1372. https://doi.org/10.1145/2470654.2466180

[11] Matthew Michael Botvinick. 2012. Hierarchical reinforcement learning and decision making. *Current Opinion in Neurobiology* 22, 6 (2012), 956–962.

[12] Daniel Buschek, Benjamin Bisinger, and Florian Alt. 2018. ResearchIME: A mobile keyboard application for studying free typing behaviour in the wild. In *CHI '18: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 14.

[13] Shi Cao, Anson Ho, and Jibo He. 2018. Modeling and predicting mobile phone touchscreen transcription typing using an integrated cognitive architecture. *International Journal of Human–Computer Interaction* 34, 6 (2018), 544–556. https://doi.org/10.1080/10447318.2017.1373463

[14] Kyle R. Cave and Narcisse P. Bichot. 1999. Visuospatial attention: Beyond a spotlight model. *Psychonomic Bulletin & Review* 6, 2 (1999), 204–223.

[15] Noshaba Cheema, Laura A. Frey-Law, Kourosh Naderi, Jaakko Lehtinen, Philipp Slusallek, and Perttu Hämäläinen. 2020. Predicting mid-air interaction movements and fatigue using deep reinforcement learning. In *CHI '20: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 13.

[16] Xiuli Chen, Gilles Bailly, Duncan P. Brumby, Antti Oulasvirta, and Andrew Howes. 2015. The emergence of interactive behavior: A model of rational menu search. In *CHI '15: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, New York, NY, 4217–4226.

[17] Xiuli Chen, Sandra Dorothee Starke, Chris Baber, and Andrew Howes. 2017. A cognitive model of how people make decisions through interaction with visual displays. In *CHI '17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 1205–1216.

[18] Jacob Cohen. 1992. A power primer. *Psychological Bulletin* 112, 1 (1992), 155–159.

[19] Richard P. Cooper. 2010. Cognitive control: Componential or emergent? *Topics in Cognitive Science* 2, 4 (2010), 598–613.

[20] Parisa Eslambolchilar and Roderick Murray-Smith. 2008. Control centric approach in designing scrolling and zooming user interfaces. *International Journal of Human–Computer Studies* 66, 12 (2008), 838–856.

[21] Anna Maria Feit, Daryl Weir, and Antti Oulasvirta. 2016. How We Type: Movement Strategies and Performance in Everyday Typing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, CA) *(CHI '16)*. ACM, New York, NY, 4262–4273. https://doi.org/10.1145/2858036.2858233

[22] Paul M. Fitts. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology* 47, 6 (1954), 381–391. https://doi.org/10.1037/h0055392

[23] Michael J. Frank and David Badre. 2011. Mechanisms of hierarchical reinforcement learning in corticostriatal circuits 1: Computational analysis. *Cerebral Cortex* 22, 3 (June 2011), 509–526. https://doi.org/10.1093/cercor/bhr114 arXiv:http://oup.prod.sis.lan/cercor/article-pdf/22/3/509/17305694/bhr114.pdf

[24] Julien Gori, Olivier Rioul, and Yves Guiard. 2018. Speed–accuracy tradeoff: A formal information-theoretic transmission scheme (Fitts). *ACM Transactions on Computer–Human Interaction (TOCHI)* 25, 5 (2018), 33.

[25] Yves Guiard, Halla B. Olafsdottir, and Simon T. Perrault. 2011. Fitt's law as an explicit time/error trade-off. In *CHI '11: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Vancouver, BC). ACM, New York, NY, 1619–1628. https://doi.org/10.1145/1978942.1979179

[26] Yves Guiard and Olivier Rioul. 2015. A mathematical description of the speed/accuracy trade-off of aimed movement. In *Proceedings of the 2015 British HCI Conference*. ACM, New York, NY, 91–100.

[27] Paul Holleis, Friederike Otto, Heinrich Hussmann, and Albrecht Schmidt. 2007. Keystroke-level model for advanced mobile phone interaction. In *CHI '07: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, CA). ACM, New York, NY, 1505–1514. https://doi.org/10.1145/1240624.1240851

[28] D.C. Howell, M. Rogier, V. Yzerbyt, and Y. Bestgen. 1998. *Statistical methods in human sciences.* Wadsworth, New York, NY.

[29] Andrew Howes. 2018. Interaction as an emergent property of a Partially Observable Markov Decision Process. In *Computational Interaction.* Oxford University Press, Oxford, UK, 287–310.

[30] Andrew Howes, Richard L. Lewis, and Alonso Vera. 2009. Rational adaptation under task and processing constraints: Implications for testing theories of cognition and action. *Psychological Review* 116, 4 (2009), 717–751.

[31] Xinhui Jiang, Yang Li, Jussi P.P. Jokinen, Viet Ba Hirvola, Antti Oulasvirta, and Xiangshi Ren. 2020. How We Type: Eye and Finger Movement Strategies in Mobile Typing. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI) *(CHI '20).* ACM, New York, NY, 1–14. https://doi.org/10.1145/3313831.3376711

[32] Jussi Jokinen. 2017. Touch screen text entry as cognitively bounded rationality. In *Proceedings of the 39th Annual Cognitive Science Society Meeting.* Cognitive Science Society, Seattle, WA, 624.

[33] Jussi P.P. Jokinen, Tuomo Kujala, and Antti Oulasvirta. 2020. Multitasking in driving as optimal adaptation under uncertainty. *Human Factors* (2020), 18. https://doi.org/10.1177/0018720820927687

[34] Jussi P.P. Jokinen, Sayan Sarcar, Antti Oulasvirta, Chaklam Silpasuwanchai, Zhenxin Wang, and Xiangshi Ren. 2017. Modelling learning of new keyboard layouts. In *CHI '17: Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, CO). ACM, New York, NY, 4203–4215. https://doi.org/10.1145/3025453.3025580

[35] Jussi P.P. Jokinen, Zhenxin Wang, Sayan Sarcar, Antti Oulasvirta, and Xiangshi Ren. 2020. Adaptive feature guidance: Modelling visual search with graphical layouts. *International Journal of Human–Computer Studies* 136 (2020), 102376.

[36] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence* 101, 1–2 (1998), 99–134.

[37] Antti Kangasrääsiö, Jussi PP Jokinen, Antti Oulasvirta, Andrew Howes, and Samuel Kaski. 2019. Parameter inference for computational cognitive models with Approximate Bayesian Computation. *Cognitive Science* 43, 6 (2019), e12738.

[38] Davis E. Kieras and Davis E. Meyer. 1997. An overview of the EPIC architecture for cognition and performance with application to human-computer interaction. *Human–Computer Interaction* 12, 4 (1997), 391–438.

[39] I. Scott MacKenzie and R. William Soukoreff. 2002. A model of two-thumb text entry. In *Graphics Interface.* Canadian Information Processing Society, Toronto, CA, 117–124.

[40] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533.

[41] Jörg Müller, Antti Oulasvirta, and Roderick Murray-Smith. 2017. Control theoretic models of pointing. *ACM Transactions on Computer–Human Interaction (TOCHI)* 24, 4 (2017), 36.

[42] Hugo Nicolau and Joaquim Jorge. 2012. Touch typing using thumbs: Understanding the effect of mobility and hand posture. In *CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM, New York, NY, 2683–2686.

[43] Donald A Norman and Tim Shallice. 1980. Attention to Action: Willed and Automatic Control of Behavior. Technical Report No. 8006.

[44] Antti Oulasvirta, Sunjun Kim, and Byungjoo Lee. 2018. Neuromechanics of a button press. In *CHI '18: Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems.* ACM, New York, NY, 13.

[45] Antti Oulasvirta, Per Ola Kristensson, Bi Xiaojun, and Andrew Howes. 2018. *Computational interaction.* Oxford University Press, Oxford, UK.

[46] Antti Oulasvirta, Anna Reichel, Wenbin Li, Yan Zhang, Myroslav Bachynskyi, Keith Vertanen, and Per Ola Kristensson. 2013. Improving Two-thumb Text Entry on Touchscreen Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris) *(CHI '13).* ACM, New York, NY, 2765–2774. https://doi.org/10.1145/2470654.2481383

[47] Kseniia Palin, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How do people type on mobile devices? Observations from a study with 37,000 volunteers. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services.* ACM, New York, NY, 12.

[48] Kseniia Palin, Anna Maria Feit, Sunjun Kim, Per Ola Kristensson, and Antti Oulasvirta. 2019. How Do People Type on Mobile Devices? Observations from a Study with 37,000 Volunteers. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei) *(MobileHCI '19).* ACM, New York, NY, Article 9, 12 pages. https://doi.org/10.1145/3338286.3340120

[49] Philip Quinn and Shumin Zhai. 2016. A Cost-Benefit Study of Text Entry Suggestion Interaction. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, CA) *(CHI '16).* ACM, New York, NY, 83–88. https://doi.org/10.1145/2858036.2858305

[50] Keith Rayner. 2009. The 35th Sir Frederick Bartlett Lecture: Eye movements and attention in reading, scene perception, and visual search. *Quarterly Journal of Experimental Psychology* 62, 8 (2009), 1457–1506.

[51] Dario D Salvucci. 2001. An integrated model of eye movements and visual encoding. *Cognitive Systems Research* 1, 4 (2001), 201–220. https://doi.org/10.1016/S1389-0417(00)00015-2

[52] Dario D Salvucci and Niels A Taatgen. 2008. Threaded Cognition: An integrated theory of concurrent multitasking. *Psychological Review* 115, 1 (2008), 101–130.

[53] Sayan Sarcar, Jussi P.P. Jokinen, Antti Oulasvirta, Zhenxin Wang, Chaklam Silpasuwanchai, and Xiangshi Ren. 2016. Towards ability-based optimization for aging users. In *Proceedings of the International Symposium on Interactive Technology and Ageing Populations.* ACM, New York, NY, 77–86.

[54] Sayan Sarcar, Jussi P.P. Jokinen, Antti Oulasvirta, Zhenxin Wang, Chaklam Silpasuwanchai, and Xiangshi Ren. 2018. Ability-based optimization of touchscreen interactions. *IEEE Pervasive Computing* 17, 1 (2018), 15–26.

[55] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[56] Miika Silfverberg, I. Scott MacKenzie, and Panu Korhonen. 2000. Predicting text entry speed on mobile phones. In *CHI '00: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM, New York, NY, 9–16.

[57] Mason R Smith, Richard L Lewis, Andrew Howes, Alina Chu, Collin Green, and Alonso Vera. 2008. More than 8,192 ways to skin a cat: Modeling behavior in multidimensional strategy spaces. In *Proceedings of the 30th annual conference of the Cognitive Science Society.* Cognitive Science Society, Austin, TX, 1441–1446.

[58] Richard S. Sutton and Andrew G. Barto. 1998. *Reinforcement learning: An introduction.* MIT Press, Cambridge, MA.

[59] Yuan-Chi Tseng and Andrew Howes. 2008. The adaptation of visual search strategy to expected information gain. In *CHI '08: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM, New York, NY, 1075–1084.

[60] Paul D. Varcholik, Joseph J. Laviola Jr, and Charles E. Hughes. 2012. Establishing a baseline for text entry for a multi-touch virtual keyboard. *International Journal of Human–Computer Studies* 70, 10 (2012), 657–672.

[61] R. William Soukoreff and I. Scott Mackenzie. 1995. Theoretical upper and lower bounds on typing speed using a stylus and a soft keyboard. *Behaviour & Information Technology* 14, 6 (1995), 370–379. https://doi.org/10.1080/01449295089146 56

[62] Jacob O. Wobbrock. 2007. Measures of Text Entry Performance. In *Text Entry Systems*, I. Scott MacKenzie and Kumiko Tanaka-Ishii (Eds.). Morgan Kaufmann, Burlington, 47 – 74. https://doi.org/10.1016/B978-012373591-1/50003-6

[63] Mingrui Ray Zhang, Shumin Zhai, and Jacob O. Wobbrock. 2019. Text entry throughput: Towards unifying speed and accuracy in a single performance metric. In *CHI '19: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems.* ACM, New York, NY, 13.