International Journal of Human–Computer Interaction

ISSN: 1044-7318 (Print) 1532-7590 (Online) Journal homepage: www.tandfonline.com/journals/hihc20

# Enhancing Mobile Multitasking with Reinforcement Learning-Based Attention Management

**Alexander Lingler, Helena Anna Frijns, Nelson J. S. Silva, Patrick Ebel & Philipp Wintersberger**

**To cite this article:** Alexander Lingler, Helena Anna Frijns, Nelson J. S. Silva, Patrick Ebel & Philipp Wintersberger (07 Jul 2026): Enhancing Mobile Multitasking with Reinforcement Learning-Based Attention Management, International Journal of Human–Computer Interaction, DOI: 10.1080/10447318.2026.2688492

**To link to this article:** https://doi.org/10.1080/10447318.2026.2688492



Published online: 07 Jul 2026.

INTERNATIONAL JOURNAL OF HUMAN–COMPUTER INTERACTION
https://doi.org/10.1080/10447318.2026.2688492



# Enhancing Mobile Multitasking with Reinforcement Learning-Based Attention Management

Alexander Lingler[a], Helena Anna Frijns[a], Nelson J. S. Silva[b,c], Patrick Ebel[d] and Philipp Wintersberger[a]

[a]Intelligent User Interfaces Group, IT:U, Linz, Austria; [b]Learning Division, IT:U, Linz, Austria; [c]IDN, Medical University of Graz, Graz, Austria; [d]Computational Interaction Group, Hasso Plattner Institute, University of Potsdam, Potsdam, Germany

**ABSTRACT**

Attention Management Systems promise to support users in better handling demands and mitigate the negative effects of multitasking. However, significant challenges remain in transferring these concepts to realistic settings. We introduce a reinforcement learning-based attention management system that schedules mobile operations with dynamic priorities, which we trained on a computationally rational simulation of typing behavior. To evaluate its effectiveness, we conducted two user studies, one in a stationary and one in a mobile context. Third, we conducted a simulation-based analysis comparing our RL-based policy with a heuristic baseline. The analysis demonstrated improved performance on several metrics for the RL-based policy as compared to self-supervision, including the number of words typed per minute and reaction times. The approach can successfully handle multiple concurrent tasks and worked in both the sitting and walking scenarios. We conclude by discussing implications for the design of Attention Management Systems.

**KEYWORDS**

Computational rationality; attention management; mobile interaction; interruptions; multitasking

## 1. Introduction

*Imagine you are inspecting a facility while carrying a handheld device. A vibration sensor reports an overheating pump, a software update request appears for another system, and a colleague sends a quality-control alert. Each task differs in priority—some must be handled immediately to avoid downtime, others can wait until later. You need to decide which task to finish first, all while moving safely through the site.*

Mobile multitasking situations are increasingly common, driven by the demands of modern, hybrid work styles. We check emails during meetings, switch songs while driving, or switch between smartphone apps while walking. Unfortunately, humans are not very good at managing multitasking demands, and frequent task switches lead to higher stress levels, error rates, and reduced performance (Bailey & Konstan, 2006; Iqbal & Bailey, 2005). Ultimately, the consequences of multitasking range from socioeconomic costs at workplaces to severe accident risks (e.g., while driving). Why is it so hard for us to manage our attention safely and productively?

To counter these effects, research has suggested developing Attention Management Systems (AMSs), which *computationally seek to balance a user's need for minimal disruption and the application's need to efficiently deliver information* (Anderson et al., 2018; Bailey & Konstan, 2006). AMSs achieve this by more efficient timing of interruptions (i.e., at task boundaries) or via resumption cues, to allow for re-attending to previously interrupted tasks more easily (Schneegass & Draxler, 2021). Still, many AMS concepts hardly benefit users and are difficult to apply to real-life settings (Anderson et al., 2018). Some systems utilize multiple machine learning models (for detecting interruptibility in users and scenarios, performing decisions, etc.) that need to be hand-crafted for each context and require large data

**CONTACT** Patrick Ebel patrick.ebel@hpi.de Computational Interaction Group, Hasso Plattner Institute, University of Potsdam, 14482 Potsdam, Germany

sets labeled by experts (Iqbal & Bailey, 2010; Kaur et al., 2020). Since multitasking behaviors are highly individual (Morgan et al., 2013), it is challenging to adapt approaches that are based on cognitive architectures for personalization (Edwards et al., 2021; Jokinen et al., 2021b). In addition, the costs of imperfect decisions, as well as the benefits of correct task switching, are not immediately visible and unfold over longer time frames, with measurable performance deficits resulting from many small, individual decisions. This raises the question of whether there is a more suitable alternative.

Reinforcement Learning (RL) is a machine learning technique that can partly handle the above-mentioned issues of limited datasets, decision-making under uncertainty, and long-term rewards. Howes et al. (2023) have argued that RL can be used to "better understand users and build better cooperative agents", and RL has demonstrated its potential to model human behavior and support human-machine cooperation (Gasse et al., 2025; Gebhardt et al., 2019; Gebhardt et al., 2020; Liao et al., 2020; Lorenz et al., 2026; Miazga et al., 2026; Xu & Zhang, 2023; Yu et al., 2022). In the context of AMS, our previous work (Lingler et al., 2024b) demonstrated that an RL multitasking agent can boost human performance. This was achieved by training this agent with an RL user model based on the concept of computational rationality (Oulasvirta et al., 2022). However, this work focused on highly artificial tasks.

Bringing AMS into real-life contexts is challenging. First, in real-world contexts, there are many concurrent activities that a user could attend to. Second, to successfully derive a policy that helps real humans, RL-based AMSs need to be (pre-)trained on user simulations in virtual task environments. A sophisticated AMS for the scenario described above would require a simulation that replicates the spatial environments, dangers, various mobile applications that generate requests with differing prioritization, options to act in this environment, and suitable functions to reward well-made sequences of actions. The simulated user, in turn, must support realistic movements, interaction with the environment and apps, and models of cognitive processes resulting from interruptions. Computationally rational models of human behavior have been demonstrated for multiple involved use cases, such as supervisory control, multitasking while driving, or attention switching with head-mounted displays (Bai et al., 2024; Gebhardt et al., 2020; Jokinen et al., 2025). While these works have confirmed that the underlying models accurately simulate human-like behavior, it is not guaranteed that cooperative agents trained on them can successfully cooperate with human users, too. In addition, real environments and human behavior differ from simulations, creating a "sim2real" gap (Dimitropoulos et al., 2022; Zhao et al., 2020). For cooperative AI, this gap is particularly large because the agent must cooperate closely with a real human who was not part of the simulation environment. Given these considerations, we address the following research questions in this work:

We ask whether an RL-based AMS can be scaled up to a higher number of concurrent tasks (RQ1); whether it also works in a scenario that is not modeled in the task environment (RQ2); and finally, if this approach can better adapt to different types of users than task-switching heuristics (RQ3).

We developed an RL-based AMS where a user has to respond to requests with dynamic priorities on a smartphone. Our AMS effectively reduces the switch costs of users and was evaluated in two user studies as well as a simulation. In summary, this paper contributes to the design of AMS as follows:

- **Concept:** We bring the concept of RL-based AMS from artificial to more realistic settings with multiple concurrent activities. We developed a task environment with the Unity3D engine that simulates text replication on a mobile device. Incoming requests have different priorities, which requires constantly re-assessing the situation to make sequential multitasking decisions.
- **Cognitive Modeling:** We developed a CR-based user model to operate in our task environment, consisting of a typing and a vision agent. This model allows the overarching AMS to learn a policy from a human-like simulation of task execution. We confirm that CR can be used not only to analyze human behavior in virtual environments but also to support real humans in complex, real-world multitasking situations. Our work extends existing models that require pre-trained transition probabilities (Jokinen et al., 2021a), as our solution dynamically updates its belief state at runtime, making our system more adaptable and extendable.
- **Evaluation:** The resulting AMS systems were evaluated in two experiments with human users and a simulation-based evaluation. In Study #1, we investigated whether the trained AMS can improve

users' performance in a stationary setting with multiple parallel requests (**RQ1**). In Study #2, we additionally burdened participants with a concurrent walking task that was not part of the environment model to investigate the sim2real gap (**RQ2**). Finally, we use a simulation to demonstrate the adaptability of our concept to different types of users (**RQ3**).

Our results show a significant performance benefit in both studies. The AMS provides task scheduling and task-switching decisions that reduce the monitoring demand and switch costs. The results of Study #2 confirm our previous results and also demonstrate that RL-based AMS systems can benefit users in a real-life setting that was not part of the environment simulation and user models. Finally, Study #3 highlights the advantages of RL-based AMS regarding performance, behavior, and adaptation in comparison to heuristic approaches. Our software, models, and the collected data are hosted in a public repository[1]. We consider our contributions an important step toward cooperative agents that support humans during multitasking.

## 2. Related work

### 2.1. Multitasking

Salvucci et al. (2009) describes multitasking as a continuum from concurrent (tasks are essentially performed simultaneously) to sequential multitasking (tasks are performed for longer durations and temporally separated). Switching between tasks includes interruptions, which can be defined as *"the suspension of one stream of work prior to completion, with the intention of resuming it later"* (Boehm-Davis & Remington, 2009). Interruptions come with intrinsic costs, which are related to the cognitive effort needed when working on multiple tasks in parallel (Fischer & Plessow, 2015; Mark et al., 2008; Salvucci & Bogunovich, 2010). Task switching can disrupt cognitive processes, leading to decreased performance, increased stress, and higher error rates (Bailey & Konstan, 2006; Borst & Taatgen, 2007; Speier et al., 1999; Wylie & Allport, 2000). Factors such as task complexity, frequency and duration of interruptions, and the nature of the secondary task influence these costs (Altmann et al., 2014; Cades et al., 2007; Couffe & Michael, 2017; Czerwinski et al., 2000; Iqbal & Bailey, 2005). An underlying key mechanism is the resumption lag, the time needed to recall the task context and reengage after an interruption (Salvucci et al., 2009). This lag is closely tied to working memory demands, as the interrupted task must be reconstructed (Trafton et al., 2003).

The timing of task switches influences the cognitive costs associated with interruptions. Research indicates that interruptions occurring at natural breakpoints, such as between subtasks, result in shorter resumption times and reduced disruption (Adamczyk & Bailey, 2004; Iqbal & Bailey, 2005; Salvucci & Bogunovich, 2010; Wintersberger et al., 2019). For instance, Monk et al. (2004) demonstrated that interruptions placed between subtasks led to significantly quicker task resumption compared to those occurring mid-subtask. However, Katidioti et al. (2014) noted that self-interruptions can be more cognitively demanding than externally imposed ones. The decision-making process involved in choosing when to switch tasks requires additional mental effort, making the act of self-interruption itself resource-intensive. Moreover, the cognitive cost of interruptions is context-dependent. Physical activity, environmental conditions, and cognitive demands can make interruptions more or less disruptive depending on complexity and problem state (Cades et al., 2007; Gillie & Broadbent, 1989; Speier et al., 2003). This is particularly relevant in mobile scenarios, where attention is already divided between digital and physical demands (Mark et al., 2005; Oulasvirta et al., 2005). Given the negative impact of interruptions on performance and stress, we investigate whether an RL-based AMS would work effectively in both stationary and mobile environments.

### 2.2. Attention management systems

AMSs aim at minimizing disruption by intelligently timing information delivery. The core idea is to consider the user's context, cognitive load, and task engagement to identify opportune moments for interruption. By postponing or advancing notifications based on such factors, AMSs reduce resumption

lag, lower stress, and improve overall task performance (Adamczyk & Bailey, 2004; Ho & Intille, 2005; Pejovic & Musolesi, 2015). The user's context can be inferred through various sensor modalities, including physiological signals or indicators of task progress (Pejovic & Musolesi, 2014; Züger et al., 2018). Machine learning models can then utilize this contextual information to predict opportune moments for task switching (Kim et al., 2018; Pielot et al., 2017; Iqbal & Sarker, 2018).

AMSs are not limited to optimizing interruption timing alone. They can also assist with task resumption by offering cues that help users reengage with previously suspended tasks. These cues may take the form of summaries of the task state at interruption time or are designed more implicitly, for instance, by guiding the user's attention toward relevant interface elements (Schneegass & Draxler, 2021; Srivastava et al., 2021). Resumption cues can be delivered before, during, or after the interruption, and can vary in modality and level of explicitness (Oulasvirta & Saariluoma, 2006).

Despite growing interest, real-time AMSs are still rare. Existing systems rely on computational models or data-driven methods (Anderson et al., 2018), but often lack flexibility or require expert-labeled data. As a result, current AMSs tend to handle interruptions in relatively simple ways, such as reacting to pre-defined task boundaries or selecting between basic modalities (Brumby et al., 2018). In prior work, we developed an RL-based system for dynamically scheduling task switches while accounting for human constraints such as visual limitations and reaction times (Lingler et al., 2024b). Unlike many data-driven approaches, this system does not rely on prelabeled expert data or training data in general. The present work builds on this idea by applying RL-based attention management to more realistic mobile multitasking contexts while considering multiple concurrent tasks.

### 2.3. Computationally rational models of human behavior

Reinforcement learning (RL) has been widely used to learn policies that allocate limited computational resources in complex systems. For example, in mobile systems research RL is applied to optimize constrained resources such as task offloading, caching, and communication scheduling in mobile edge computing environments (Mao et al., 2016a; Yang et al., 2025a; Yang et al., 2025b). Computational rationality (CR) offers a promising framework for modeling human multitasking behavior under uncertainty and individual variability (Oulasvirta et al., 2022). Within this framework, RL is used to learn policies that allocate limited cognitive resources, such as attention, memory, and motor control, across competing tasks while respecting human constraints. Unlike classical RL, which typically aims for superhuman performance, CR incorporates human-like constraints to produce behavior that aligns with human capabilities. A key distinction in CR is the separation between an internal and external environment. The internal environment represents the agent's cognitive state, shaped by limitations related to visual perception, memory, or motor control (Oulasvirta et al., 2022). The external environment consists of observable stimuli and task dynamics. The transition from the external to the internal environment, based on noisy observations, results in an imperfect internal representation of the external state. The transition is modeled using partially observable Markov decision processes (POMDPs) (Barto et al., 2004), where the agent acts based on a belief state. The internal model can include psychological factors such as memory decay, noise, or stress, and the rational policy is defined as the optimal action given these cognitive limitations (Howes et al., 2023). One key advantage of CR is its adaptability. By parameterizing cognitive constraints, user models can be adjusted to reflect both general and individual behavior. This allows for more realistic and personalized simulations.

Recent research has applied CR to model touchscreen typing behavior, focusing on the coordination of visual attention and motor actions. For instance, Jokinen et al. (2021a) conceptualize touchscreen typing as an optimal supervisory control problem, where users learn to manage their visual and motor resources to maximize typing performance. Building upon this, Shi et al. (2024) introduce *CRTypist*, a model that simulates human-like typing behavior by integrating visual attention and motor control, operating directly from pixel data without manual feature engineering. An extension of this approach models the cognitive mechanisms behind typing errors, including slips, lapses, and mistakes, by simulating eye and finger movements to produce human-like error patterns (Shi et al., 2025).

Building upon prior research, we implemented a CR model of touchscreen typing in Unity using the AITentive toolkit (Lingler et al., 2024a) to train our AMS, enhancing its ability to predict optimal task-

switching moments. Thereby, our approach integrates concurrent tasks scheduled by an AMS as a cooperative agent.

## 3. Attention management in the mobile context

In this work, we investigated how an RL-based AMS can schedule mobile operations with different priorities. The AMS monitors the list of incoming requests, as well as the user, and optimizes task switches so that overall performance increases. For training, we implemented a CR typing model with human vision. The system supports both *forced* interruptions (automated switching) and *suggestions* (task switches are indicated, but a user can decide whether to follow them). Figure 1 provides an overview of the proposed RL-based attention management system. Prioritized tasks occur in many mobile work contexts: logistics and delivery workers receive time-critical updates about new jobs or route changes, field technicians handle incoming maintenance alerts with predefined urgency levels, and care staff respond to requests with varying priority. Priorities in these settings are typically assigned by the underlying system rather than inferred from user input, making them well suited for an AMS that schedules attention across concurrent tasks.

### 3.1. The task environment: Mobile operations with dynamic priorities

To investigate our research questions, we introduce a conceptual mobile operations task that serves as a proxy for broader mobile interactions, allowing the AMS to optimize attention shifts dynamically (see Figure 2). It can be deployed in mobile contexts where users frequently face both sequential and concurrent multitasking demands.

Therefore, we employed a text-replication task to capture typical mobile input behavior as present in communication applications. Communication represents the largest category of smartphone interactions (Böhmer et al., 2011), and text entry is pervasive across a wide range of mobile activities, including messaging, search, note-taking, or interactions with language models (Chen et al., 2025). Recent studies further show that users continue to rely on typing despite the availability of alternative input modalities such as speech (Luo et al., 2022). Moreover, typing involves a sequence of target acquisition, motor execution, visual monitoring, and error correction, reflecting core interaction primitives common to many smartphone tasks (Feit et al., 2016). This task was also selected due to the availability of CR models capturing these perceptual-motor processes (Jokinen et al., 2021a; Shi et al., 2024), whereas models for more complex cognitive decision-making remain limited, making model availability the main constraint for extending our approach. Finally, text replication provides a controlled abstraction that enables objective measurement of typing performance across both computational models and human participants (MacKenzie & Soukoreff, 2002). Given these considerations, we argue that our proxy task accounts for a high and representative share of interactions performed on mobile devices.

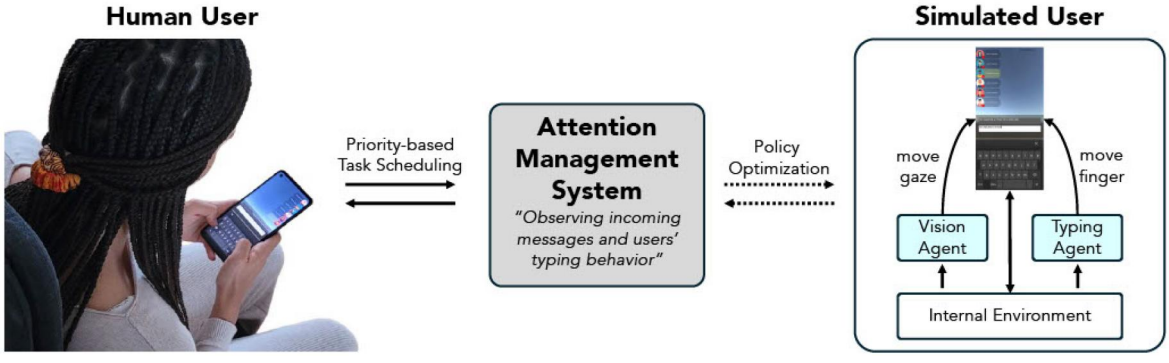



**Figure 1.** We Propose an RL-based attention management system that schedules mobile operations with dynamic priorities. By repeatedly interacting with simulated users grounded in computationally rational models of typing, the system learns to optimize task timing and minimize disruptions.

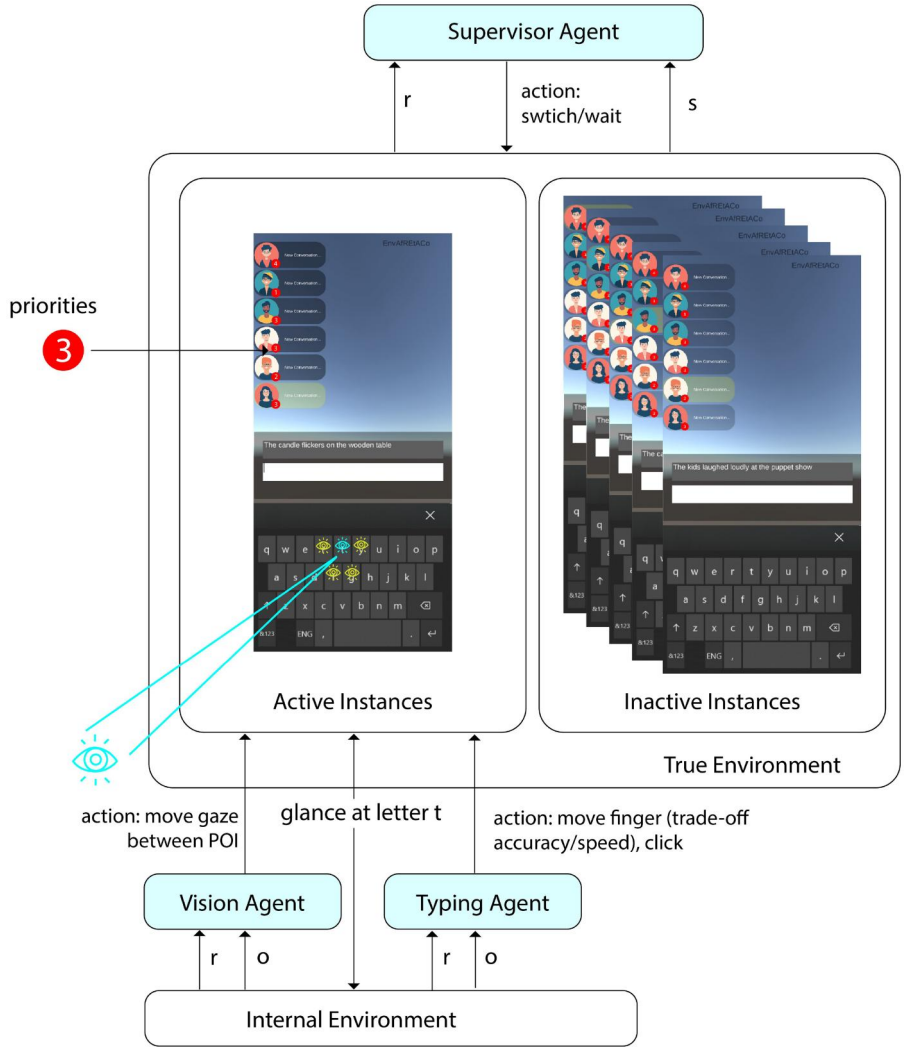



**Figure 2.** The environment's architecture consists of three distinct agents: the *supervisor agent (AMS)*, the *typing agent*, and the *vision agent*. The AMS has access to the true environment state, while the typing and vision agents operate within an internal environment, which the vision agent is responsible for updating *(r: reward, o: observation, s: state)*.

The task environment includes up to 6 requests that can be active in parallel (as we wanted to avoid scrolling, this limitation emerged from the screen size of the mobile device and not the AMS concept). Each request has a priority level from 0 to 5 (indicated by the number within the red circle in Figure 2). A request is completed after the enter button is pressed. The incoming requests are generated based on a configuration file that specifies their priority, the delay before they appear, the total number of requests, and the maximum number of concurrent open requests. If this maximum is reached, a new request will only be activated when another request is completed. A trial is completed after replicating all requests. Users have to respond to requests considering their priority, typing accuracy, and speed. Whenever users switch between requests, their progress and content are saved.

The environment was implemented in Unity and uses the ml-agents library (Juliani et al., 2018) and the AITentive toolkit (Lingler et al., 2024a). A trial run starts with a 15-second countdown. Initially, the environment loads only a single request and gets populated according to the configuration file, with newer requests appearing at the top. Additionally, priorities of non-selected requests can change over time to simulate dynamic multitasking scenarios with varying demands. Selected requests are marked with a yellow background.

In the *force* mode, the AMS automatically switches between the requests, indicated by a multimodal interruption. The combination of an auditory signal and the screen turning white announces a task

switch 0.5 s before it occurs. This was added to alert users about the upcoming switch so they had the opportunity to prepare. In *suggestion* mode, requests are marked in two colors: yellow indicates the user's selection, while red represents the AMS's suggestion. When the user follows the suggestion, the color turns green. The keyboard featured haptic and visual feedback for button presses.

### 3.2. Partially observable Markov decision processes

In RL, an agent in state $s$ performs an action $a$, leading to a transition to $s'$ with probability $p$ and receiving reward $r$. The agent learns a policy $\pi$ (with) that maximizes expected return. In our setup, the AMS agent has access to the true state, yielding a fully observable $MDP(S, A, P, R, \gamma)$, where the discount factor $\gamma \in [0, 1]$ controls how future rewards are weighted. *Throughout, uppercase symbols (e.g., S, A, P, R, Ω) denote the corresponding sets of elements—states, actions, transition probabilities, rewards, and observations, respectively (analogously for the POMDP tuple)—while $\gamma$ is a scalar discount factor.* Value-based RL estimates the state-value function via the Bellman expectation equation:

$$v_\pi(s) = \sum_{a \in A} \pi(a|s) \sum_{s' \in S} \sum_{r \in R} p(s', r|s, a)\left[r + \gamma \; v_\pi(s')\right] \quad \text{for all } s \in S. \tag{1}$$

Computing $v_\pi(s)$ is recursive over successor states and rewards; because the state space in this work is continuous, function approximation (e.g., neural networks) is used to derive actions in that space.

By contrast, the typing and vision agents cannot fully observe the environment: only the portion attended by the vision agent is observed and updated, while unobserved parts increase uncertainty and can degrade action quality. Such settings are modeled as a $POMDP(S, A, P, R, \gamma, H, \Omega, O)$ with an observation set $\Omega$ and observation function $O$, where $O(s', o, a)$ gives the probability of observing $o$ in state $s'$ after action $a$ (computed here using a fixed parameter $o_p$, see Formula 6). Since the true state is hidden, decisions are made over a belief state $b(s)$, a distribution over $S$, updated as:

$$b'(s') \propto O(s', o|s, a) \sum_{s \in S} p(s'|a, s) b(s), \tag{2}$$

The proportionality is chosen so that $\sum_{s'} b'(s') = 1$. Policies in the POMDP are therefore defined over beliefs $b$ rather than true states $s$.

### 3.3. AMS architecture: Optimizing response time and typing performance

The AMS was trained using a state space of six requests and must determine the best time to switch tasks based on the dynamic priorities and task states. Therefore, the AMS has access to the full state space of all concurrent tasks (see Figure 2), i.e., the true environment state, as its goal is to learn a task-switching policy that supports human performance.

Given the dynamic prioritization, simple heuristics, such as always completing a response before switching, may not be sufficient. More sophisticated rules (i.e., depending on typing speed or priority differences), in turn, would add another layer of participant-dependency, while RL can address tradeoffs resulting from individual differences and dynamic adaptations in a state- and user-dependent manner.

To train the AMS, we implemented typing agents that mimic human behavior. This was achieved by incorporating human constraints into their RL environment, specifically accounting for motor noise and cognitive limitations. The CR agents do not have direct access to the true environment; instead, they operate on an internal environment modeled as a POMDP, including the estimated finger position and perceived written answer (see Figure 2). The vision agent continuously updates its belief by refining either the finger position or the written answer, simulating human proofreading. In the following, the implementation of the individual agents and system components is described in detail.

### 3.4. AMS agent

The AMS agent $\kappa_a$ is responsible for switching between requests to maximize a given reward. Due to display restrictions, no more than six requests are shown at a time. Each request is either active

(priority 1–5, with 5 the highest) or idle (priority 0, indicating no response required). Each episode comprises 25 requests in total; when a request is answered, the next one is drawn from the pool until all 25 are completed. To prevent switching to an idle request, we applied action masking.

The AMS environment is modeled as an MDP where $S_a$ contains all request states. Here, $s \in S_a$ is defined by the last button pressed by the typing agent, the typing duration $t_d$, and the estimated completion ratio of the written answer, given by $\frac{L_w}{L_g}$, where $L_w$ represents the length of the written answer and $L_g$ is the estimated length of the target sentence. Additionally, the state includes the priority $\phi_q$ of the request. The action $a \in A_a$ enables the agent to either switch to an active request or remain on the current one. The agent makes a decision every second, and an additional decision is triggered whenever a request is fully answered. $p(s'|s,a) \in P_a$ represents the probability of transitioning to the successor state $s'$ after performing action $a$ in state $s$. In the forced condition of the experiment, the transition probability for switching to a specific request is deterministic, meaning that after performing action $a$, the agent will certainly switch to the target request. However, this is not the case for the suggestion scenario, where the user can still choose to follow their own policy. Additionally, the state space concerning the requests is stochastic, as another agent is responsible for handling the mobile operations task, introducing state space manipulations that are not controlled by the AMS. The reward $r \in R_a$ is defined by the following formula:

$$r = \sum_{q \in Q} r_q \cdot \phi_q \tag{3}$$

Therefore, the reward is the sum over all rewards $r_q$ of the different requests $q \in Q$ multiplied by their priorities $\phi_q$. Here, $r_q$ is defined as follows:

$$r_q = 2 \cdot (L_g - lev(g,w)) - t_d \tag{4}$$

Here, $lev(g,w)$ represents the *Levenshtein* distance between the goal sentence and the written sentence. This linear formulation allows for both positive and negative rewards, where the possibility of negative penalties helps prevent request "starvation". If a metric like words per minute were used, delaying engagement with the request would lead to an artificially low WPM, even if the agent eventually types very quickly, contradicting the intended priority system.

In our experimental setup, the model used both the participant's written response and the predefined target text to estimate task progress. While this relies on ground-truth information unavailable in real applications, it serves as a controlled proxy for incremental task completion. In practical systems, comparable progress measures could be derived from observable indicators—such as completed form fields in data entry, distance remaining in navigation, or verified subtasks in maintenance. Thus, although text comparison is domain-specific, the underlying concept of progress-based feedback generalizes to a wide range of mobile task environments.

### 3.5. Typing and vision agents

The AMS is trained using a typing agent that integrates the principles of CR into a typing task. We extend the approach presented in (Jokinen et al., 2021a) by incorporating their proofreading agent into our typing agent model and adapting both the reward function and the state representation. Additionally, instead of pretraining transition probabilities, the belief state is updated dynamically at runtime. The typing agent's internal belief is continuously updated by a vision agent, creating a dynamic interaction that models realistic human behavior. The environment contains the following human constraints:

- Internal representation of the believed written text;
- Internal representation of finger position;
- Finger movement constrained by human-like speed and natural movement noise, based on the weighted homographic (WHo) model of pointing (Guiard & Rioul, 2015);
- Eye movement between points of interest (POI), following the EMMA model (Salvucci, 2001);
- Encoding of elements near the fovea, as described by the EMMA model.

#### 3.5.1. Motor control

The typing agent $\kappa_\tau$ can be modeled as a POMDP. The belief state $b(s)$ represents the estimated finger position on the keyboard, and is therefore a probability distribution over $S_\tau$. To model this, the keyboard was discretized into 3,000 bins, each assigned a probability indicating the likelihood of the finger being in that position. The transition probability $p \in P_\tau$ in Equation 2 is updated based on the finger landing variability described by the WHo model:

$$(y - y_0)^{1-\alpha}(x - x_0)^{\alpha} = k_\alpha \tag{5}$$

Here, $x$ denotes mean finger movement time ($\mu_T$) and $y$ expresses relative landing variability. Following the original WHo formulation, $y$ is the relative endpoint spread $\sigma_A/\mu_A$, where $\sigma_A$ is the standard deviation of landing locations across repetitions and $\mu_A$ is the traveled distance (movement amplitude). In our notation we therefore set $\sigma \equiv \sigma_A$ and use $y = \sigma/\mu_A$ to connect our runtime variability estimate to the WHo model. The constants $\alpha$ and $k_\alpha$ shape the curve. Precise movements (small $y$) take longer, while faster movements (small $x$) increase the finger landing variability, which scales with movement amplitude. Therefore, the bin probabilities are updated at runtime based on the standard deviation $\sigma$. A low $\sigma$ (slow finger movement) concentrates probabilities in a focused area, while a high $\sigma$ (fast movement) spreads them across many bins. The probability distribution update is computed in parallel and independently for each bin using the *Unity C# Job System* combined with the *Burst Compiler* (Ferreira & Geig, 2018). The state space $S_\tau$ further contains information about the belief target button (depending on the written text of belief) and the entropy of $b(s)$. The observation probability $O_\tau$ in Eq. (2) is determined using a fixed parameter $o_p$:

$$O_\tau(s', o|s, a) = \begin{cases} o_p & \text{if } s' \text{ is the true new state and the gaze is on the finger;} \\ \dfrac{1 - o_p}{N_s - 1} & \text{if } s' \text{ is not the new true state and the gaze is on the finger;} \\ 1 & \text{if the gaze is not on the finger.} \end{cases} \tag{6}$$

where $N_s$ is the number of states and $0 \leq o_p \leq 1$ holds. This observation probability enables a fast belief update to the true state if the bin where the finger is located is encoded by the vision agent. If the bin is not encoded, no observation occurs, s.t. $O_\tau(s', o|s, a) = 1$ for all $s' \in S_\tau$. The action $a \in A_\tau$ involves moving the finger to a button or pressing it while setting $\sigma$ to balance speed and accuracy in movement. The typing reward $r_{\text{type}}$ is defined as follows:

$$r_{\text{type}} = \begin{cases} 0.5 \cdot (N_c + 1) & \text{If, according to the belief, the correct button was pressed and it is not the delete button;} \\ 0.5 & \text{If, according to the belief, the correct button was pressed and it is the delete button;} \\ -0.5 & \text{if the wrong button was pressed.} \end{cases} \tag{7}$$

where $N_c$ describes the number of correct written letters according to the belief. Therefore, in the first case of this reward structure, the more letters that are written correctly, the higher the rewards. This approach discourages agents from answering the request before it is fully completed. The remaining cases address incorrectly written letters, ensuring that the agent can return to its original reward if a wrongly written letter is deleted. In case the vision agent looks at the written text, the accumulated reward $r_t \in R_\tau$ at time step $t$ of the episode is updated based on the proofreading reward $r_{\text{pr}}$ by

$$r_t = -r_{\text{pr}} + \sum_{i=1}^{t} r_{\text{type}}(i) \tag{8}$$

$r_{\text{pr}}$ is defined as follows:

$$r_{\text{pr}} = f(C_{\text{last}}) - f(C_{\text{current}}) - 0.5 \cdot N_{\text{del}} \tag{9}$$

where $C_{\text{last}}$ represents the number of correctly written letters in the previously proofread text, and $C_{\text{current}}$ denotes the same for the current text. $N_{\text{del}}$ defines the number of necessary deletions to obtain the target text. The function $f$ returns the collected reward of the first case of the formula of $r_{\text{type}}$ based on the number of correctly written letters and is defined as:

$$f(x) = \frac{x}{2} \cdot \left( \frac{x}{2} + 1.5 \right) \tag{10}$$

which is just the partial sum of $a_n = a_{n-1} + 0.5$ with $a_0 = 0$. Thus, the proofreading reward grants a reward of 0.5 for each correctly written letter and deducts the same amount for each incorrectly written letter. Consequently, if the agent writes only incorrect letters, the previously awarded reward $r_{\text{type}}$, which was given based on its belief, is subtracted again. This structure mirrors human behavior by separating typing and proofreading while ensuring that the typing and vision agents share a common goal. At the same time, the vision agent can independently earn rewards for proofreading. This setup naturally encourages periodic text verification, supports finger positioning, and maintains efficient training.

### 3.5.2. Human vision

The vision agent $\kappa_v$ is responsible for updating the belief state of the typing agent by moving its gaze between POIs, i.e., the buttons of the keyboard and the written text, and is defined as a MDP. The state space $S_v$ is defined based on the entropy of $b(s)$ as defined in Section 3.5.1, the number of text input entries not verified by proofreading, and the target letter of the belief according to the written text. The entropy of $b(s)$ reflects the vision agent's certainty about the finger position, while the unverified text entries indicate its confidence in the written text. Using this information, the agent should develop a policy that balances checking the finger position and verifying the text. An action $a \in A_v$ involves shifting the gaze between POIs based on the EMMA model, which defines the relationship between information encoding and eye movement. The time required to encode a visual object is defined as

$$T_{\text{enc}} = K \cdot \left[ -\log(f) \right] \cdot k^{k\epsilon} \tag{11}$$

where $K$ and $k$ are encoding constants, $f$ is the frequency of the object, and $\epsilon$ is the eccentricity (distance from fixation). With static object frequencies, the visual system can reduce eccentricity and encoding time by making a saccade that lasts:

$$T_s = t_{\text{prep}} + t_{\text{exec}} + D \cdot t_{\text{sacc}} \tag{12}$$

where $t_{\text{prep}}$, $t_{\text{exec}}$, and $t_{\text{sacc}}$ are constants, and $D$ is the saccade distance in degrees. If $T_{\text{enc}} < t_{\text{prep}}$, encoding occurs without eye movement. Otherwise, a saccade shifts the fixation closer to the target before completing the encoding. Since the vision and typing agents share a common goal, they share the same reward signal $r$ as defined in equation 8 and therefore $R_v = R_\tau$.

## 4. Training

Training was conducted using the ML-Agents Unity framework, with all agents optimized via Proximal Policy Optimization (PPO). In the first stage, the typing and vision agents were trained independently of the AMS. Each agent was trained over 100 million steps, where one step corresponds to a single decision, and both reached convergence within this duration.

The vision agent employed the following constants according to (Salvucci, 2001): $t_{\text{prep}} = 0.135, \text{s}$, $t_{\text{exec}} = 0.07, \text{s}$, $t_{\text{sacc}} = 0.002, \text{s}$, $K = 0.006$, $k = 0.4$, and $f = 0.1$. The WHo model parameters of the typing agent followed prior work simulating touchscreen pointing (Jokinen et al., 2021a), using $\alpha = 0.6$, $k_\alpha = 0.12$, $x_0 = 0.092$, and $y_0 = 0.0018$. The decision request interval for each agent was determined by its respective model.

In the second stage, the AMS was trained for 20 million steps, with each decision request occurring every 1 s based on the pretrained vision and typing models, achieving convergence after this period.

## 5. User studies

We assessed the AMS through two distinct user studies conducted in sitting and walking contexts, as well as an evaluation in simulation. The research proposal was reviewed by the ethics committee at Interdisciplinary Transformation University Austria, under case number 2025-10. All participants provided informed consent prior to participation.

### 5.1. Study 1: Mobile multitasking while sitting

Participants completed typing tasks while seated in each of the following conditions:

- **Force**: Task switches are enforced by the AMS without user intervention. The switches can be performed at any time.
- **Suggestion**: The AMS switches are only suggested. Participants were informed that they can decide whether they follow the suggestion immediately.
- **No Supervisor**: The AMS was absent, requiring participants to manage the switching of requests themselves (baseline condition).

We designed three distinct mobile multitasking scenarios (i.e., configurations), each defining the total number of requests to be solved per condition, the interval between request appearances, their sequence, priority, and the maximum number of open requests at any given time. Each scenario consisted of 25 requests with an average text length of 35.92 characters ($SD = 2.23$), ranging from 30 to 42 characters. The average priority across scenarios was 3, and these scenarios were systematically varied between the conditions. We decided to use predefined scenarios rather than random ones, as many performance variables are heavily influenced by the request priority and the order of appearance. To evaluate performance, we measured a wide range of variables:

- **Duration**: Total time required to solve 25 requests.
- **Gross Words per Minute**: Calculated as $\frac{\left(\frac{N_{\text{keystrokes}}}{5}\right)}{T_{\text{con}}}$, where $N_{\text{keystrokes}}$ represents the total number of keystrokes per condition and $T_{con}$ denotes the duration per condition in minutes.
- **Net Words per Minute**: Defined as $\text{GWPM} - \frac{lev}{T_{\text{con}}}$, where *lev* represents the *Levenshtein* distance between the written and the target sentence.
- **Keystroke Error Rate**: Ratio between number of errors and total number of keystrokes $\frac{N_{\text{error}}}{N_{\text{total}}}$.
- **Priority/Accuracy Weighted Gross Words per Minute**: Computed as *GWPM* multiplied by the request priority and its accuracy.
- **Supervisor Reward** $r_q$ : Accounts for request starvation as specified in Eq. (4).

After each condition, workload was assessed using the short version of the NASA-TLX questionnaire (Hart & Staveland, 1988). Additionally, participants rated their overall experience on a single 7-point Likert scale item (“*Please rate your overall experience of playing under this condition, from* “don’t like at all” *to* “like it very much”“). Furthermore, we analyzed the switching behavior between conditions for behavioral differences:

- **Reaction Time**: the duration between a task switch and the first subsequent key press.
- **Intermediate Interruptions**: the count of task switches occurring after initiating a request, but before completing it.
- **Intermediate Interruptions (%)**: proportion of *Intermediate Interruptions* relative to the total number of task switches.
- **Interruption Levenshtein Distance**: the Levenshtein distance between the written and target text at the point of an *Intermediate Interruption*.

To analyze the results across the three conditions, we used IBM SPSS V.29 to perform Friedman tests with subsequent pairwise Bonferroni-corrected Wilcoxon tests or repeated measures ANOVA with post-hoc tests, depending on whether the data followed a normal distribution.

#### 5.1.1. Participants

Overall, $N = 21$ participants (12 female, 9 male, $M = 29.38, SD = 7.6$ years, mainly students and university staff) completed the experiment. Of the participants, 19 use their smartphone on a daily basis. Most of the participants use smartphones and laptops to solve their typing tasks. For typing, 18 use both fingers while 3 use one. Their self-rated proficiency with a touchscreen on a Likert scale from 1 to 4 was $M = 2.81$ ($SD = 0.75, Median = 3$). Their self-rated typing speed on a Likert scale from 1 to 5 was $M = 3.38$ ($SD = 0.80, Median = 3$). Participants' tendency toward actively engaging with technology, as measured by the Affinity for Technology Interaction (ATI) scale (Franke et al., 2019), was slightly above neutral ($M = 4.20, SD = 1.42, \alpha = 0.91$).

#### 5.1.2. Procedure

The experiment was carried out directly within the Unity editor, and the data was streamed to a Galaxy S10 phone via Wi-Fi with the help of the Modoium app, leading to a small input delay. As can be seen in Figure 2, the keyboard was a custom implementation, giving haptic and visual feedback when pressing a button. Upon arrival, participants were informed about the study procedure, followed by signing the consent form and a short training session to get familiar with the interface and the keyboard. Next, participants completed the three conditions in a quasi-randomized order, with quasi-randomized configuration assignment. Participants were informed that the experiment would end automatically once all requests were solved. The exact number was unknown to them. A post-condition survey was completed after each condition. At the end, data collection was complemented by a post-experiment survey. In total, the experiment took approximately 45 min per participant.

#### 5.1.3. Results

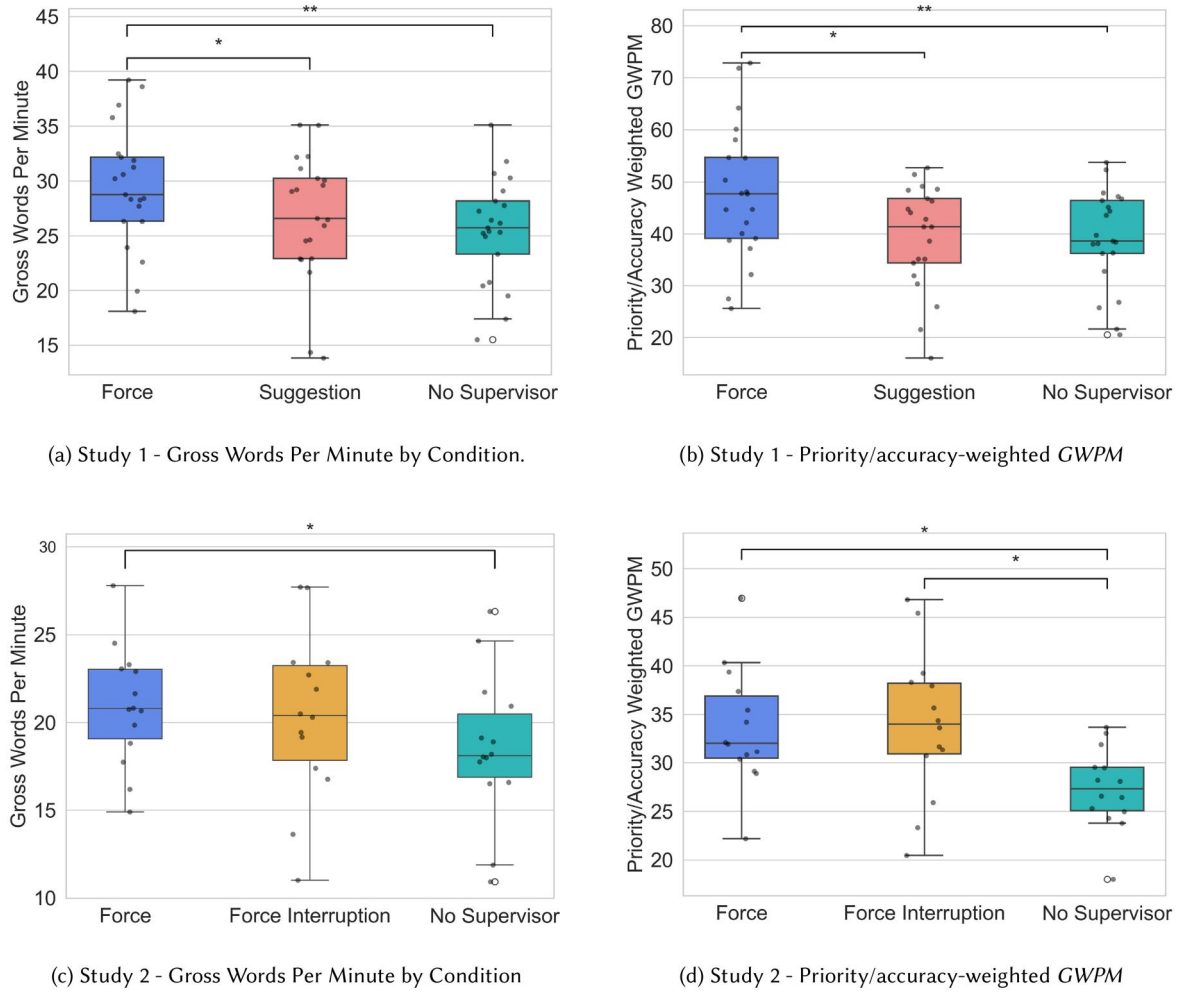
***Performance.*** Table 1 presents descriptive statistics of participants' absolute performance across the individual conditions as well as workload ratings. Participants' *GWPM* and priority/accuracy-weighted *GWPM* for Study 1 are shown in Figure 3a and 3b. An analysis of participants' *GWPM* across the three conditions in Study 1 (*force*, *suggestion*, and *no supervisor*) revealed a significant effect (Greenhouse-Geisser; $F(1.383, 27.668) = 15.00,\ p < .001,\ \eta_p^2 = .429$). Bonferroni-corrected pairwise comparisons showed that performance in the *force* condition was significantly higher than in both the *suggestion* ($p = .025$) and *no supervisor* ($p < .001$) conditions.

A Friedman test assessing *NWPM* differences among the three conditions yielded a significant result ($\chi^2(2) = 20.67,\ p < .001$). Post hoc analyses indicated that *NWPM* was significantly lower in the *no*

**Table 1.** Participants' performance and subjective ratings in **study 1** (M (SD)) where conditions were *force*, *suggestion*, and *No supervisor*.

| | Force | Suggestion | No supervisor |
|---|---|---|---|
| Performance data | | | |
| GWPM | **29.43 (5.60)** | 26.71 (5.76) | 25.55 (4.83) |
| NWPM | 26.62 (6.96) | 24.90 (6.22) | 23.40 (6.03) |
| Duration | 237.60 (53.16) | 250.71 (62.37) | 254.61 (48.41) |
| Keystroke error rate | 0.0870 (0.0379) | 0.0767 (0.0356) | 0.0817 (0.0388) |
| Priority/accuracy GWPM | **47.75 (12.96)** | 39.41 (10.00) | 39.09 (9.39) |
| Supervisor reward | 39.76 (46.48) | 24.76 (59.51) | ***19.65 (45.43)*** |
| Behavior | | | |
| Reaction time | 1.542 (0.336) | 1.421 (0.334) | **1.854 (0.328)** |
| Int. interruptions | 5.05 (2.459) | 2.76 (0.994) | **1.14 (2.081)** |
| Int. interruptions (%) | 16.10 (7.27) | 10.41 (2.96) | **5.14 (8.03)** |
| Interruptions Lev. distance | 21.97 (9.68) | 22.94 (4.13) | 28.38 (5.39) |
| Subjective workload | | | |
| Mental demand | 3.33 (1.461) | 3.24 (1.578) | 4.62 (1.499) |
| Physical demand | 3.05 (1.802) | 2.81 (1.289) | 3.24 (1.700) |
| Temporal demand | 4.05 (1.717) | 3.86 (1.389) | 4.43 (1.434) |
| Effort | 3.57 (1.690) | ***3.48 (1.365)*** | 4.76 (1.300) |
| Frustration | 4.48 (1.601) | 3.71 (1.521) | 4.57 (1.434) |
| Subj. performance | 4.52 (1.167) | 5.05 (1.322) | 4.24 (0.995) |
| User Preference | | | |
| Overall rating | 3.71 (1.586) | **5.10 (1.179)** | 4.00 (1.265) |

Values marked in **bold** indicate that significant differences were found, while values that are marked in ***bold italic*** indicate that a significant value was found but not with respect to both conditions; please refer to the main body text for details.




(a) Study 1 - Gross Words Per Minute by Condition.

(b) Study 1 - Priority/accuracy-weighted *GWPM*

(c) Study 2 - Gross Words Per Minute by Condition

(d) Study 2 - Priority/accuracy-weighted *GWPM*

**Figure 3.** Selected performance metrics illustrating overall typing speed (*GWPM*) and priority/accuracy-weighted *GWPM* across all conditions in both studies. Higher values indicate better performance. * significance level $p < 0.05$, ** significance level $p < 0.001$.

*supervisor* condition compared to the *suggestion* condition ($p = .010$) and the *force* condition ($p < .001$).

Regarding the total duration across conditions, a Friedman test did not reveal a significant difference ($\chi^2(2) = 5.43$, $p = .066$). Similarly, no significant differences were found in the keystroke error rate across conditions ($\chi^2(2) = .29$, $p = .867$). A repeated measures ANOVA examined the effect of condition on priority/accuracy-weighted *GWPM*. Mauchly's Test of Sphericity was not significant ($W = .812$, $\chi^2(2) = 3.96$, $p = .138$), confirming that the assumption of sphericity was met. The results indicated a significant main effect of condition ($F(2, 40) = 14.48$, $p < .001$, $\eta_p^2 = .420$). Pairwise comparisons with Bonferroni correction revealed that participants performed significantly better in the *force* condition than in both the *suggestion* ($p = .002$) and *no supervisor* ($p < .001$) conditions. Finally, a Friedman test evaluating supervisor reward differences among conditions indicated a significant effect ($\chi^2(2) = 6.381$, $p = .041$). Post hoc pairwise comparisons with Bonferroni correction revealed that supervisor reward was significantly lower in the *no supervisor* condition compared to the *force* condition ($p = .041$).

***Request selection and switching behavior.*** A Friedman test revealed a significant effect of condition on reaction time, $\chi^2(2) = 23.47$, $p < .001$. Pairwise comparisons indicated significantly longer reaction times in the *no supervisor* condition compared to both *suggestion* ($p < .001$) and *force* ($p = .001$). Similarly, the number of intermediate switches differed significantly across conditions, $\chi^2(2) = 21.95$, $p < .001$, with fewer switches in the *no supervisor* condition than in the *suggestion* ($p = .013$) and *force* ($p < .001$) conditions. This suggests that participants without supervisory support were more likely to

complete the current text before switching tasks. Notably, only 8 participants performed intermediate switches.

When examining the proportion of intermediate switches relative to the total number of switches, results also showed a significant effect ($\chi^2(2) = 14.63$, $p < .001$), with the *no supervisor* condition again showing a significantly lower proportion compared to *suggestion* ($p = .028$) and *force* ($p = .001$). Additional indicators of task boundaries, such as space bar presses (signaling word completion), revealed that unsupervised users were more likely to switch after finishing a word (33.34%) than those in the *force* (25.26%) or *suggestion* (29.31%) conditions. Finally, regarding progress on the left request at the time of switching, no significant effect was found, $\chi^2(2) = 3.68$, $p = .159$. Although the mean progress differed between the *force* ($M = 21.97$, $SD = 9.68$) and *no supervisor* ($M = 28.25$, $SD = 5.39$) conditions, the analysis was limited to $N = 8$ participants who actually performed intermediate switches.

***Subjective workload and experience.*** For *Effort*, Friedman's test showed a significant difference, $\chi^2(2) = 11.246$, $p = .004$. Wilcoxon tests indicated lower effort in *suggestion* than *no supervisor* ($p = .010$). An uncorrected trend suggested lower effort in *force* vs. *no supervisor* ($p_{unadj} = .031, p = .092$). Friedman tests indicated significant differences among conditions for *Mental demand* ($\chi^2(2) = 7.159$, $p = .028$) and *Subjective performance* ($\chi^2(2) = 6.185$, $p = .045$). However, pairwise Wilcoxon tests with Bonferroni correction found no significant differences (all $p > .05$). Uncorrected trends suggested higher *Mental demand* in *no supervisor* than *force* ($p_{unadj} = .025, p = .076$), and higher *Subjective performance* in *suggestion* than *no supervisor* ($p_{unadj} = .037, p = .112$). For *Frustration*, *Physical demand*, and *Temporal demand*, no significant differences were indicated.

For the overall rating, Friedman's test was significant, $\chi^2(2) = 8.827$, $p = .012$. Wilcoxon tests showed higher ratings in *suggestion* than *force* ($p = .041$) and *no supervisor* ($p = .050$).

### 5.2. Study 2: Mobile multitasking while walking

In this study, participants solved the typing task while walking. Figure 4 shows a picture of the walking path that had to be completed. Again, participants completed three conditions, two with AMS assistance and one without. Both AMS conditions utilized the forced mode, as the suggestion mode did not result in improved performance in Study 1 (see Section 5.1.3). The AMS variants in the study conditions can be distinguished based on the training approach:

- **Force:** In this condition, the model used is the same as the one used in the first user study.
- **Force-Interruption (Force Int.)**: This condition used an AMS for which the agent was trained using an environment that included interruptions, as measured in the work of Steil et al. (2018). To do this, the data was filtered for instances of using WhatsApp in a street walking context. The data was then sampled during training, defining how long the typing task remained active and how long it

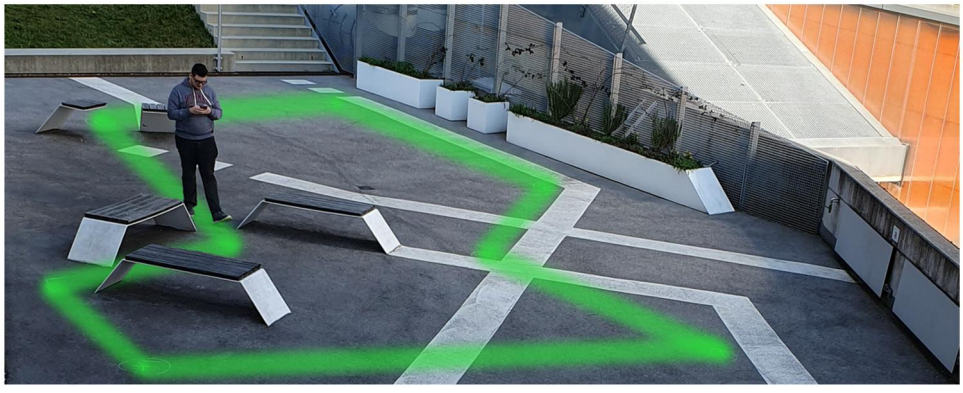


**Figure 4.** The green line depicts the prescribed walking path for participants. All corners were chalk-marked. The environment contained no moving pedestrians or traffic; benches were the only static obstacles.

was interrupted. The AMS cannot suggest switches between the walking and typing tasks, as the walking environment is unknown to it. However, based on interruptions related to scanning the street environment, the agent learns that these interruptions could present an opportune moment for a task switch. The agent detects interruptions based on the time of the last action performed, which is added to the state space $S$.
- **No Supervisor:** In this condition, the AMS was absent, similar to Study 1.

In addition to the performance metrics of Study 1, we measured the number of laps completed, and eye-tracking data to measure the human gaze behavior shifting between the typing task and the surrounding:

- **Gaze on Phone** (%): percentage of fixations directed at the phone, relative to the total number of fixations.
- **Pupil Size**: average pupil diameter and its standard deviation, measured in millimeters.
- **Gaze Dispersion**: computed as $\sqrt{\sigma_x^2 + \sigma_y^2}$, where $\sigma_x$ and $\sigma_y$ represent the standard deviations of fixation movements along the horizontal (x) and vertical (y) axes, respectively, in pixels.

#### 5.2.1. Participants

$N = 15$ participants (12 female, 3 male, age $M = 30.40, SD = 6.00$ years, mainly students and university staff) completed the experiment. All participants use their smartphone on a daily basis. Most of the participants use smartphones and laptops to solve their typing tasks; 12 use both fingers for typing while 2 use one, and 1 uses another technique. Their self-rated proficiency with a touchscreen on a Likert scale from 1 to 4 was $M = 2.80$ $(SD = 0.676, Median = 3)$. Their self-rated typing speed on a Likert scale from 1 to 5 was $M = 3.6$ $(SD = 0.632, Median = 4)$. Participants' tendency toward actively engaging with technology, as measured by the Affinity for Technology Interaction (ATI) scale (Franke et al., 2019), was slightly above neutral ($M = 4.42, SD = 1.17, \alpha = 0.835$).

#### 5.2.2. Procedure

The procedure closely followed that of the first user study, with the following exceptions. As shown in Figure 4, the study took place outdoors on an asphalted square with no traffic or pedestrians. After the initial briefing, the eye tracker was set up and calibrated with the participants. A *Pupil Labs Neon* eye tracker was used to measure eye movements. During the training session, participants were required to walk one lap of the course while typing. After the training, participants started at position 1 with an initial 15-second countdown and navigated through a path with seven obstacles or junctions. The condition ended once all 25 requests had been processed. Consequently, there was no predefined maximum number of laps. Instead, the number of laps completed was recorded.

#### 5.2.3. Results

***Performance.*** Table 2 shows the results of study 2, including the eye tracking results. The corresponding performance metrics for Study 2 are shown in Figure 3c and 3d. A series of Friedman tests was conducted to assess differences across the three conditions. For *GWPM*, the test yielded a significant result, $\chi^2(2) = 6.53$, $p = .038$. Post hoc analyses indicated that *GWPM* was significantly lower in the *no supervisor* condition compared to the *force* condition ($p = .032$).

No significant differences were observed for *NWPM*, $\chi^2(2) = 4.80$, $p = .091$, total duration, $\chi^2(2) = 2.80$, $p = .247$, or keystroke error rate, $\chi^2(2) = 0.400$, $p = .819$. For priority/accuracy-weighted *GWPM*, a significant effect was observed, $\chi^2(2) = 10.13$, $p = .006$. Post hoc tests revealed significantly lower values in the *no supervisor* condition compared to both the *force* ($p = .032$) and *force interruption* conditions ($p = .010$). Finally, supervisor reward also differed significantly across conditions, $\chi^2(2) = 6.93$, $p = .031$, with the *no supervisor* condition showing significantly lower values than the *force* condition ($p = .032$).

**Table 2.** Participants' performance and subjective ratings in **study 2** (M (SD)) where conditions were *force*, *force int.*, and *No supervisor*.

| | Force | Force int. | No supervisor |
|---|---|---|---|
| Performance data | | | |
| GWPM | 22.06 (5.47) | 21.58 (6.54) | ***20.69 (9.26)*** |
| NWPM | 18.10 (5.26) | 17.88 (5.33) | 16.91 (6.86) |
| Duration | 286.32 (60.50) | 308.68 (95.53) | 314.72 (124.33) |
| Keystroke error rate | 0.0826 (0.0283) | 0.0914 (0.0415) | 0.0921 (0.0429) |
| Priority/accuracy GWPM | 35.36 (8.96) | 35.61 (9.78) | **28.25 (5.21)** |
| Supervisor reward | −19.72 (52.37) | −33.75 (72.51) | ***−60.62 (79.47)*** |
| Behavior | | | |
| Reaction time | 1.88 (0.392) | 1.69 (0.273) | **2.39 (0.424)** |
| Int. interruptions | 5.60 (2.444) | 13.27 (10.02) | **1.20 (1.897)** |
| Int. interruptions (%) | 17.83 (6.37) | **31.80 (13.69)** | 5.62 (8.67) |
| Interruptions Lev. distance | 29.16 (5.86) | 22.59 (5.39) | 19.87 (13.35) |
| Laps/minute | 2.015 (0.586) | 2.086 (0.672) | 2.064 (0.687) |
| Eye tracking | | | |
| Gaze on phone (%) | 91.65 (9.94) | 91.81 (7.71) | 90.77 (9.86) |
| Pupil size (M) | 2.64 (0.311) | 2.61 (0.325) | 2.59 (0.316) |
| Pupil size (SD) | 0.192 (0.051) | 0.197 (0.061) | 0.194 (0.056) |
| Gaze dispersion (M) | 6.02 (2.11) | 6.68 (2.86) | 5.69 (1.027) |
| Gaze dispersion (SD) | 1.53 (0.677) | 1.94 (1.67) | 1.49 (0.363) |
| Subjective workload | | | |
| Mental demand | 3.73 (1.280) | ***3.33 (1.718)*** | 4.53 (1.727) |
| Physical demand | 4.00 (1.309) | 4.00 (1.195) | 4.13 (1.846) |
| Temporal demand | 3.67 (1.496) | 3.87 (1.767) | 4.00 (1.690) |
| Effort | 4.07 (1.387) | 4.00 (1.852) | 4.53 (1.727) |
| Frustration | 4.00 (1.558) | 3.53 (1.995) | 4.27 (1.668) |
| Subj. performance | 4.33 (0.816) | 4.40 (1.121) | 4.07 (0.704) |
| User Preference | | | |
| Overall rating | 4.33 (1.447) | 4.53 (1.246) | 3.60 (1.242) |

*Force int.* describes the model trained within the environment that included interruptions.

***Request selection and switching behavior.*** A repeated measures ANOVA assessed the effect of condition on reaction time. Mauchly's Test of Sphericity was not significant ($W = .866$, $\chi^2(2) = 1.88$, $p = .391$), confirming the assumption of sphericity. The ANOVA revealed a significant main effect, $F(2, 28) = 21.07$, $p < .001$, $\eta_p^2 = .60$, indicating differences in reaction time across conditions. Bonferroni-corrected comparisons showed significantly longer reaction times in the *no supervisor* condition compared to both *force* ($p = .005$) and *force interruption* ($p < .001$).

The number of intermediate switches also differed significantly, $\chi^2(2) = 29.53$, $p < .001$, with fewer switches in the *no supervisor* condition than in the *force interruption* ($p < .001$) and *force* ($p = .032$) conditions. Additionally, participants switched less frequently in the *force* condition compared to *force interruption* ($p = .014$). Notably, only 7 participants made intermediate switches. A significant effect was also found for the proportion of intermediate switches relative to total switches, $\chi^2(2) = 28.13$, $p < .001$, with the highest proportion observed in the *force interruption* condition compared to *force* ($p = .010$) and *no supervisor* ($p < .001$). Participants in the *no supervisor* condition were more likely to switch after completing a word (33.34%) than those in the *force* (28.57%) and *force interruption* (18.04%) conditions.

A significant effect was found for progress on the left request at the time of switching, $\chi^2(2) = 7.14$, $p = .028$. The Levenshtein distance between the written and target word was significantly lower in the *force interruption* condition compared to *force* ($p = .023$). Finally, a repeated measures ANOVA assessed the effect of condition on the laps per minute. Mauchly's Test of Sphericity was not significant ($W = .841$, $\chi^2(2) = 2.25$, $p = .325$), confirming the assumption of sphericity. The ANOVA revealed no significant main effect, $F(2, 28) = .583$, $p = .565$, $\eta_p^2 = .040$.

***Eye tracking.*** A Friedman test revealed a significant difference in the average gaze dispersion, $\chi^2(2) = 7.385$, $p = .025$. However, pairwise Wilcoxon tests with Bonferroni correction found no significant differences. An uncorrected trend suggested a difference between *force interruption* and *force* ($p_{unadj} = .019, p = .056$) and *force interruption* and *no supervisor* ($p_{unadj} = .019, p = .056$). For the other eye-tracking measures, no significant differences between conditions were found.

***Subjective workload and experience.*** A Friedman test revealed significant differences in *Mental demand* across conditions, $\chi^2(2) = 8.68$, $p = .013$. Post hoc analyses with Bonferroni correction showed significantly lower *Mental demand* in the *force interruption* condition compared to *no supervisor* ($p = .024$). No significant differences were found for *Subjective performance*, *Frustration*, *Physical*, or *Temporal demand*. A repeated measures ANOVA assessed the effect of condition on *Effort*. Mauchly's Test of Sphericity was significant ($W = .506$, $\chi^2(2) = 8.85$, $p = .012$), violating the assumption of sphericity; therefore, Greenhouse-Geisser correction was applied. The ANOVA revealed no significant effect of condition on *Effort*, $F(1.34, 18.74) = 1.29$, $p = .284$, $\eta_p^2 = .084$.

For overall rating, Mauchly's test indicated that sphericity was not violated ($W = .762$, $\chi^2(2) = 3.53$, $p = .172$). The ANOVA showed a significant main effect of condition, $F(2, 28) = 3.62$, $p = .040$, $\eta_p^2 = .205$, although Bonferroni-corrected pairwise comparisons did not reveal any significant differences.

## 6. Study 3: Simulation – sensitivity to typing speed

To examine whether our AMS adapts to individual differences in skill levels, we conducted a simulation-based analysis comparing our RL-based policy with a heuristic baseline. In the proxy task, the skill level corresponds to typing speed. We used the same mobile operations environment and reward structure introduced earlier, clustered participants of Study #1 by words-per-minute (WPM) into three groups (*slow*, *medium*, *fast*), and parameterized the typing models by modifying the parameters $x_0$, $y_0$, $\alpha$ and $k_\alpha$ of Eq. (5). For each group, we trained a separate RL-based AMS policy and compared it with a heuristic baseline that always switches to the request with the currently highest priority. We chose this greedy baseline because it represents a transparent, priority-driven strategy that requires no parameter tuning (as discussed in section 3.3) and serves as a clear lower bound for adaptive switching behavior. In practice, priority changes occurred infrequently in our study, so the greedy heuristic did not lead to excessive switching. We report *Intermediate Interruption* and *Supervisor Reward* $r_q$ in Table 3.

### 6.1. Adaptive switching behavior

Table 3 shows that the RL-based AMS adjusted its switching policy across typing-speed groups, producing marked differences in the *Number of Intermediate Interruptions*. Paired-samples $t$ tests (heuristic vs. RL within each group) show fewer interruptions under the RL policy for all groups (all $p < .001$). For slow typists, the reduction was significant ($\Delta = 3.82$, $t(49) = 13.32$, $p < .001$, $d = 1.88$); for medium typists, likewise ($\Delta = 4.02$, $t(49) = 17.26$, $p < .001$, $d = 2.44$); and for fast typists, the reduction was largest ($\Delta = 5.14$, $t(49) = 16.49$, $p < .001$, $d = 2.33$).

Across groups, interruption counts for the three RL policies differed significantly, indicating robust adaptation. For the heuristic, Welch's ANOVA showed a group effect, $F(2, 96.687) = 8.301$, $p < .001$, with Games–Howell tests indicating significant differences between slow and medium ($p = .004$) and between slow and fast ($p = .001$), but not between medium and fast ($p = .656$). For the RL-based AMS, Welch's ANOVA also revealed a strong group effect, $F(2, 94.842) = 72.498$, $p < .001$, and Games–Howell confirmed that all pairwise comparisons were significant (all $p < .001$). The findings regarding the adaptability of the different RL policies are summarized in Figure 5.

**Table 3.** Simulation results by typing-speed group (mean with standard deviation).

| | # Intermediate interruption | | Supervisor reward $r_q$ | |
|---|---|---|---|---|
| Group | Heuristic | RL-based | Heuristic | RL-based |
| Slow | 7.38 (1.537) | 3.56 (1.327) | 13.77 (2.06) | 15.95 (5.19) |
| Medium | 6.40 (1.443) | 2.38 (0.805) | 26.31 (2.31) | 28.56 (3.77) |
| Fast | 6.10 (1.940) | 0.96 (0.903) | 41.95 (3.40) | 44.82 (2.58) |

Lower is better for *intermediate interruption*; higher is better for *supervisor reward* $r_q$.

### 6.2. Performance

Table 3 also reports *Supervisor Reward* ($r_q$), showing that the RL-based AMS outperformed the heuristic across all typing-speed groups. For slow typists, a paired-samples $t$-test indicated a significant improvement, $t(49) = 2.76$, $p = .008$, mean difference = 2.17 [0.59, 3.76], $d = 0.39$ ($g = 0.39$). For medium typists, a paired-samples $t$-test indicated a significant improvement, $t(49) = 3.73$, $p < .001$, mean difference = 2.25 [1.04, 3.46], $d = 0.53$ ($g = 0.52$). For fast typists, a paired-samples $t$-test likewise showed a significant improvement, $t(49) = 5.06$, $p < .001$, mean difference = 2.87 [1.73, 4.01], $d = 0.72$ ($g = 0.71$).

## 7. Discussion

All three experiments' outcomes show that RL-based AMS led to significant improvements in performance measurements (i.e., words per minute, priority-weighted typing speed) compared to self-supervised manual task-switching and the heuristic evaluated in simulation. Self-supervision led to significantly longer reaction times, highlighting the effort required to pick the most appropriate task to continue. The AMS, providing *suggestions* instead of *forced* switches, increased performance only slightly. In the sitting condition (Study #1), participants favored the suggestion mode, which combined the freedom to choose tasks with a clear recommendation. The performance gain was also visible during walking (Study #2). This highlights that RL-trained AMS policies can remain beneficial in environmental conditions that are not explicitly modeled. In both experiments, participants handled up to six concurrent requests with different priorities, and the *forced* AMS led to increased performance across multiple measurements. This shows that RL-based AMS can indeed scale up to a higher number of tasks than in previously explored dual-task settings **(RQ1)**. Regarding the sim2real gap, the results indicate that the AMS maintained its advantage in contexts not modeled during training. The performance increase was also visible when participants had to follow a path. In the walking study, the AMS also reduced participants' subjective mental demand significantly, which was not visible in the static condition. Thus, the learned policy was robust enough to handle environmental changes, or at least did not degrade performance when additional requirements were imposed **(RQ2)**. The results obtained in the simulation (Study #3) further demonstrate the RL-based AMS's adaptability to users with varying capabilities **(RQ3)**.

### 7.1. Forced switching vs. user control

Our results are consistent with our previous work, which showed that forced task switches can outperform self-initiated multitasking strategies (Lingler et al., 2024b). Previous research has indicated that self-interruptions can be more disruptive than external interruptions, since self-interruptions create additional cognitive demand (Katidioti et al., 2016), which is removed by the AMS in the *forced* condition. Our findings translate well from more continuous and safety-critical processes like driving (Ebel et al., 2023; Wintersberger et al., 2018) to the investigated mobile operations task. Consequently, optimal task-switching policies are not always intuitive to users, who tend to follow a less performance-oriented approach by switching less frequently, in particular when task priorities vary or new tasks arrive unpredictably (Anderson et al., 2018). This is visible by inspecting when and how often task switches occurred. With the *forced* AMS, users switched more in general and interrupted an ongoing reply more frequently. In contrast, participants frequently ignored *suggestions*, which negatively impacted various performance measurements. Behavioral data show that participants often delayed acting on suggestions until their current task was completed and only switched to the suggested task afterwards. Therefore, the ignored suggestions reflect task-completion preferences rather than weak visual cues or disagreement with the priority assessment. In that regard, using AI to investigate and optimize task switching can help test cognitive theories. While literature often focuses on interruption/resumption lags (Iqbal & Horvitz, 2007), the significant amount of selection costs (i.e., which task to attend) should not be neglected (Trafton et al., 2003). In our studies, users' performance was more influenced by the latter. The AMS removed the burden of selection from users while sometimes slightly increasing the number of switches, compared to self-supervision. Thus, the pure task switching costs had less impact on overall performance than the task selection costs.

Still, participants tended to rate the experience with the *suggestion* mode higher, which highlights that users desire to stay in control. Despite potential performance gains, too frequent or aggressive switching may decrease the user experience, leading to frustration. Future work should focus on a more user-friendly design for forced task switches (i.e., better announcements or resumption cues) or employ a mixed-initiative approach (Horvitz, 1999). Interestingly, participants were less worried about the forced switches in the walking condition. This confirms existing literature showing higher acceptance for automation and less sense of agency in high-stakes or cognitively demanding scenarios (Dewey, 2023; Drnec et al., 2016; Kool et al., 2010; Talypova et al., 2026). Consequently, future AMS should (1) adapt their interruption mode (i.e., suggestions vs. automated switches) according to contextual factors like cognitive load, stress, or environmental demands, while (2) allowing users to configure their preferences for these modes.

### 7.2. AMS while walking: No or detailed environment models?

In Study #2, we compared the original AMS model with another policy that models context shifts derived from mobile interaction studies. Both AMS policies significantly improved participants' performance. Regarding gross words per minute, the original AMS performed even better. The *interruption* model aimed at simulating real-world distractions while walking, but did not provide advantages. Quite the opposite, the *interruption* model even led to a significantly higher number of task switches, which must be considered a negative result when no performance benefit is achieved.

One possible explanation is that the pre-defined walking path was too predictable and did not require continuous environmental monitoring. This aligns with findings from eye-tracking data, which show that users focused on the smartphone most of the time. Therefore, the underlying assumption that users would need to monitor the environment and temporarily shift their visual attention away from the smartphone was not met. As a result, the expected interruptions caused by scanning the surroundings did not occur. Consequently, the AMS, which was trained to exploit such environment-induced interruptions as opportune moments for task switching, could not demonstrate any benefits in this simplified walking scenario. Less predictable distractions on the walking path may lead to different results. Future work could explore more complex walking environments with varying demands, where the interruption model may offer more benefits. In such environments, safety-critical situations should be explicitly represented as part of the environment. Based on sensor and contextual signals (e.g., obstacles or traffic), the policy could then learn to suppress or pause interactions in high-risk situations instead of exploiting interruptions for task switching.

Nevertheless, the results also indicate that the policy, even when trained in a controlled environment that does not include the additional task, can sufficiently handle additional demands. We consider this a relevant finding with respect to the sim2real gap, since precise modeling of the walking task would also require equipping users with additional sensors that allow assigning them to the state space of a hypothetical walking environment.

### 7.3. Sensitivity to typing speed

We have argued that the demonstrated potential of CR/RL to model diverse user capabilities in a theoretically grounded manner (Jokinen et al., 2021a; Jokinen et al., 2021b; Shi et al., 2024) provides benefits for the development of cooperative AIs such as AMS, too. Our simulations show that the RL-based AMS adapts its switching policy to users' text-entry skills. As illustrated in Figure 5, the staircase traces plot priority against cumulative duration, the "X" markers denote intermediate switches and the numbers highlight certain events described in the following. The plot shows a single episode with the same order of priorities, time between requests and roughly similar texts. For fast typists, the AMS policy more often deferred a switch until the current reply was completed (no intermediate switches visible), whereas for slow typists it switched earlier to higher-priority requests, as completion would require more time and thus lead to lower expected returns (1). This pattern indicates that the policy is sensitive to both expected time-to-completion and request priority, rather than applying a fixed threshold. Across conditions, the AMS produced fewer intermediate switches than the heuristic baseline, reducing

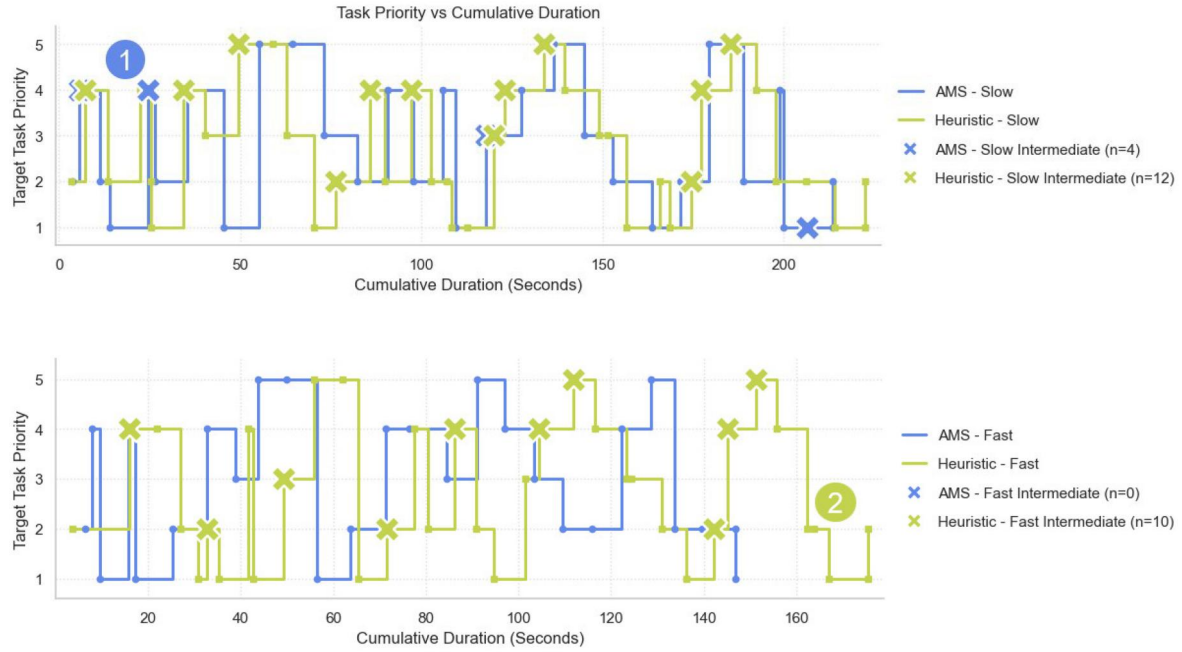



**Figure 5.** Priority vs. cumulative duration for a single episode constructed with the same priority order and inter-arrival times across conditions (texts of comparable length). The staircase traces show priority; "X" markers denote intermediate switches, while dots indicate switches after completing a request. (1) For slow typists, the AMS switches earlier to higher-priority requests, whereas for fast typists it defers until completion. (2) Relative to the heuristic baseline, the AMS yields fewer intermediate switches and a shorter overall episode duration.

disruptive mid-request switches while improving the overall objective by minimizing the overall duration compared to the heuristic. This is important, since prior work has shown that reducing the number of interruptions helps minimize performance decline and errors (Bailey & Iqbal, 2008; Anderson et al., 2018; Bailey & Konstan, 2006). In practice, this suggests that static heuristics can misfit different users, whereas policies that estimate typing speed and anticipated remaining time can better minimize interruption costs. A future AMS could adapt its switching policy to each user's skill level by relying on dynamic measures of task performance. This online adaptation could be accomplished by maintaining multiple AMS variants trained with CR-based user models representing different user profiles. During runtime, behavioral metrics such as gaze patterns, typing dynamics, or response timing could be used in combination with Bayesian inference to continuously estimate the user's CR parameters and select the AMS policy whose underlying user model best matches the observed behavior. These estimates can be derived from observable performance signals. Such measures could include typing speed in our proxy task, or alternative performance indicators in other mobile contexts, and could be estimated automatically and recalibrated as performance changes.

### *7.4. Progress and effort estimation*

In our current implementation, task progress is used to compute the reward signal and forms part of the observation space of the AMS. Depending on specific real-world applications, task progress may not be available. Closing this additional sim2real gap will require task progress estimation based on observable contextual and interaction signals. Still, many practical tasks inherently provide progress indicators (such as spatial distance or clearly defined subtasks). Predicting task progress and execution times with high accuracy is a research problem in its own. In scenarios where dynamic tasks are often highly similar, clearly defined, and conducted repeatedly (such as present in industrial environments, hospitals, or in the context of human-robot collaboration), existing research has already demonstrated the potential of supervised learning approaches (De Lazzari et al., 2025; Stisen et al., 2017). Still, even open-ended text composition could utilize statistical methods to infer task progress based on parameters such as task description, inferred structure of incoming tasks, potential responses, operating system

properties, or user context. For example, an additional component could use LLMs to create a distribution of probable continuation sequences, which are continuously updated based on the users' inputs, while CR typing models could estimate their remaining durations. Such estimates carry significant uncertainties and are a source of additional noise, which reinforces the suitability of RL-based solutions to learn effective prioritization policies, since RL is better suited to dealing with dynamic uncertainties than static heuristics (Sutton & Barto, 2018).

### 7.5. Toward general attention management

We argue that our work is another ingredient for realizing sophisticated AMS in a wide range of real settings, but multiple aspects need to be considered. First, task prioritization is a significant challenge, which we simplified for the presented experiment. Still, priorities could be implicitly or explicitly learned via the reward function (Mao et al., 2016b; Mao et al., 2019), yet real-world tasks often involve complex contextual factors (Lingler et al., 2024b; Lingler et al., 2025). Considering the presented task environment, there is potential to utilize Large Language Models (LLMs) that interpret task context or textual input fields to infer priorities directly. For our studies, we assigned and visually communicated priorities directly to not disadvantage participants in self-supervision. Inferring priorities from contextual task information using an LLM, as similarly explored for interruptibility estimation, could be a valuable additional component for adaptive task scheduling and might yield further improvements (Lingler et al., 2026).

It is important to note, however, that in many real mobile task scenarios—such as logistics coordination, emergency response, or maintenance workflows—priorities are typically predefined by the system or operational context. In such cases, the inference of priority, for example through LLMs, would not be required. In this sense, our simulated environment serves as a controlled proxy for such externally prioritized tasks, allowing us to isolate and evaluate the attention management mechanisms independently of the prioritization process.

Similarly, while our experimental setup required comparing the written response to a predefined target text to estimate progress, real-world task progress is often more directly measurable. In mobile operations, completion can typically be inferred from observable indicators such as task status updates, sensor feedback, or spatial progress toward a goal. This makes progress tracking in many applied scenarios more straightforward than in our simulated text-entry task, while the underlying reward formulation remains transferable. At the same time, a real-life AMS must accommodate a broad spectrum of mobile tasks, where priorities may arise from dynamic contextual factors or be defined externally by the system.

There are already efforts to solve computer tasks (such as the Miniwob++ benchmark) generally via LLM/RL hybrid approaches (Thil et al., 2024). Provided the problem state and goal of an arbitrary task can be estimated, progress estimation could be implemented using generative AI in combination with more traditional cognitive architectures (Knowles et al., 2023; Romero et al., 2023). Cognitive models that can simulate memory-based tasks and decision-making are important for the future of RL-based AMS in general. Most CR/RL models so far simulate skill-based activities that continuously provide actions and are observable (just as the vision and motor control model used in this work). Extending this concept to arbitrary tasks requires models to support more complex cognitive processes present in real-life task execution. Modeling these deeper cognitive aspects of human behavior, such as memory processes or reasoning over time, is a significant challenge for the next generation of computationally rational models. This limits our current AMS implementation to purely skill-based tasks while urging the need to extend behavior models to complex decision-making processes.

Further, we want to emphasize that we do not claim that RL is the only approach to improve performance in our investigated setting. Potentially, a well-crafted, more sophisticated heuristic than the one we used for our simulative-based evaluation might yield similar performance. Still, prior research has shown that rule-based, heuristic, or supervised methods tend to plateau in performance and fail to adapt to user and task variability (Horvitz et al., 2003; Hudson et al., 2003; Okoshi et al., 2015). In contrast, RL policies emerge from direct interaction with the simulated environment, and while heuristics become unmanageable across many different tasks, an RL-based AMS can easily be trained for other contexts. Finally, attention management is not only about task performance. Future work should investigate multiple objectives (for example, users' stress levels) and combine them accordingly, using multi-objective RL (Felten et al.,

2023; Hayes et al., 2022). We argue that when these issues are adequately addressed, AMS solutions can be effectively deployed to assist users in a wide range of real-world scenarios.

### 7.6. Limitations

The sample sizes were quite small, and our participants were mainly students and staff of a technical university, which limits the generalization across age groups or educational characteristics. Potentially, they had a more positive attitude toward AI systems in general. Future work should therefore include larger-scale studies with more diverse participant populations beyond academic environments to better assess the generalizability of our findings. Another limitation is the simplified smartphone task of text replication. Future work need to allow for more natural responses (i.e., question/answer settings). We decided to use text replication to have clear measurements for error rates. In the future, we will update our models using LLMs to run our user simulation, roughly estimate task progress, and calculate typing errors afterward. Further, mobile users interact with a mix of applications and interaction modalities (i.e., calls, voice assistants, social media, etc.), which need to be accounted for. Regarding the walking experiment, our pre-defined simple path did not include dynamic distractions, as present in real urban environments. We aim for more ecologically valid experiments in more complex spaces in the future. From a system perspective, we refer to the issues of task prioritization and progress estimation discussed above and also want to mention potential user adaptations over time. We pre-trained our system, yet in reality, users will adapt their own policies when working with AMS systems, which requires looking into concepts such as lifelong learning to continuously adapt to changing user behavior (Gheibi & Weyns, 2022). Finally, AMSs have ethical implications regarding the context of use and user agency. Existing research has suggested that users may welcome AMS that helps them multitask and avoid digital distractions (Talypova et al., 2023). However, these studies asked potential users at a more abstract level and did not address questions such as whether they would accept a forced mode over suggestions.

## 8. Conclusion

In this paper, we introduced a Reinforcement Learning-based Attention Management System (AMS) for scheduling multiple concurrent mobile operations tasks on mobile devices. We used a computationally rational user simulation to train our AMS to optimize task switching decisions in a way that considers dynamic priorities and evaluated the system in two user studies and in simulation. In both user studies, the AMS led to significant performance improvement compared to purely manual switching. The studies demonstrate improvements in terms of priority-weighted speed and words per minute. These benefits also extend to a walking scenario, which the AMS was not particularly trained for. In the simulation-based analysis, the RL-based policy showed fewer interruptions and resulted in higher supervisor rewards. Thus, we argue that RL-based AMS has the potential to work in contexts not taken into account during training, which may partly address the sim2real gap when bringing cooperative agents into the real world. Still, further work is needed to refine automated prioritization and progress estimation across different mobile task environments and to extend AMS models toward richer cognitive processes. At the same time, many real mobile systems already provide explicit priorities and clear task states, easing integration. Ultimately, our results advance the development of practical attention management systems for real-world use.

## Note

1. https://github.com/reapphil/AITentiveMultitasking

## Acknowledgments

This manuscript was proofread and linguistically refined using the generative AI tool ChatGPT 5. The tool was used solely to improve clarity and readability. All scientific content, analyses, and conclusions are the authors' own.

## Author contributions

CRediT: **Alexander Lingler**: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing; **Helena Anna Frijns**: Investigation, Supervision, Visualization, Writing – review & editing; **Nelson J. S. Silva**: Data curation, Resources, Validation, Writing – review & editing; **Patrick Ebel**: Formal analysis, Visualization, Writing – original draft; **Philipp Wintersberger**: Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – original draft, Writing – review & editing.

## Disclosure statement

The authors declare no competing interests.

## Funding

This project is supported by the Austrian Science Fund (FWF) under grant Nr.P35976-N (AITentive).

## ORCID

Alexander Lingler http://orcid.org/0009-0004-9439-7375
Helena Anna Frijns http://orcid.org/0000-0003-3866-7380
Nelson J. S. Silva http://orcid.org/0000-0002-4994-0963
Patrick Ebel http://orcid.org/0000-0002-4437-2821
Philipp Wintersberger http://orcid.org/0000-0001-9287-3770

## Data availability statement

The data, code, and trained models that support the findings of this study are openly available in the Open Science Framework at https://doi.org/10.17605/OSF.IO/8TM6K, reference number 8TM6K.

## References

Adamczyk, P. D., & Bailey, B. P. (2004). *If not now, when?: The effects of interruption at different moments within task execution* [Paper presentation]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Vienna, Austria) (CHI '04)., ACM, New York, NY, 271–278. https://doi.org/10.1145/985692.985727

Altmann, E. M., Trafton, J. G., & Hambrick, D. Z. (2014). Momentary interruptions can derail the train of thought. *Journal of Experimental Psychology: General*, *143*(1), 215. https://doi.org/10.1037/a0030986

Anderson, C., Hübener, I., Seipp, A.-K., Ohly, S., David, K., & Pejovic, V. (2018). A survey of attention management systems in ubiquitous computing environments. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 2, Article 58 (July 2018), 27 pp. https://doi.org/10.1145/3214261

Bai, Y., Ikkala, A., Oulasvirta, A., Zhao, S., Wang, L. J., Yang, P., & Xu, P. (2024). Heads-up multitasker: Simulating attention switching on optical head-mounted displays. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. ACM, 1–18.

Bailey, B. P., & Iqbal, S. T. (2008). Understanding changes in mental workload during execution of goal-directed tasks and its application for interruption management. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *14*(4), 1–28. https://doi.org/10.1145/1314683.1314689

Bailey, B. P., & Konstan, J. A. (2006). On the need for attention-aware systems: Measuring effects of interruption on task performance, error rate, and affective state. *Computers in Human Behavior*, *22*(4), 685–708. https://doi.org/10.1016/j.chb.2005.12.009

Barto, A. G., Singh, S., Chentanez, N., et al. (2004). Intrinsically motivated learning of hierarchical collections of skills. In *Proceedings of the 3rd International Conference on Development and Learning*, Vol. 112. Citeseer, UCSD Institute for Neural Computation, 19.

Boehm-Davis, D. A., & Remington, R. (2009). Reducing the disruptive effects of interruption: A cognitive framework for analysing the costs and benefits of intervention strategies. *Accident Analysis & Prevention*, *41*(5), 1124–1129. https://doi.org/10.1016/j.aap.2009.06.029

Böhmer, M., Hecht, B., Schöning, J., Krüger, A., & Bauer, G. (2011). Falling asleep with Angry Birds, Facebook and Kindle: A large scale study on mobile application usage. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*. ACM, 47–56.

Borst, J. P., & Taatgen, N. A. (2007). The costs of multitasking in threaded cognition. In *Proceedings of the Eighth International Conference on Cognitive Modeling*. Psychology Press, 133–138.

Brumby, D. P., Janssen, C. P., Kujala, T., & Salvucci, D. D. (2018). Computational models of user multitasking. In *Computational Interaction*. Oxford University Press. https://doi.org/10.1093/oso/9780198799603.003.0013 arXiv:https://academic.oup.com/book/0/chapter/297822125/chapter-pdf/40117570/oso-9780198799603-chapter-13.pdf

Cades, D. M., Davis, D. A. B., Trafton, J. G., & Monk, C. A. (2007). Does the difficulty of an interruption affect our ability to resume?. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 51. SAGE Publications Sage CA, 234–238. https://doi.org/10.1177/154193120705100419

Chen, W., Shi, Y., Wang, Y., Shi, W., Chen, M., Gao, C., Yu, M., Zhu, Y., Zhang, J., & Yu, C. (2025). Investigating context-aware collaborative text entry on smartphones using large language models. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. ACM, 1–20.

Couffe, C., & Michael, G. A. (2017). Failures due to interruptions or distractions: A review and a new framework. *The American Journal of Psychology*, *130*(2), 163–181. https://doi.org/10.5406/amerjpsyc.130.2.0163

Czerwinski, M., Cutrell, E., & Horvitz, E. (2000). Instant messaging: Effects of relevance and timing. In *People and Computers XIV: Proceedings of HCI*, Vol. 2. Springer, 71–76.

De Lazzari, D., Terreran, M., Giacomuzzo, G., Jain, S., Falco, P., Carli, R., Ghidoni, S., & Romeres, D. (2025). Real-time human progress estimation with online dynamic time warping for collaborative robotics. *Frontiers in Robotics and AI*, *12*, 1623884. https://doi.org/10.3389/frobt.2025.1623884

Dewey, J. A. (2023). Cognitive load decreases the sense of agency during continuous action. *Acta Psychologica*, *233*, 103824. https://doi.org/10.1016/j.actpsy.2022.103824

Dimitropoulos, K., Hatzilygeroudis, I., & Chatzilygeroudis, K. (2022). A brief survey of sim2real methods for robot learning. In *International Conference on Robotics in Alpe-Adria Danube Region*. Springer, Springer, 133–140.

Drnec, K., Marathe, A. R., Lukos, J. R., & Metcalfe, J. S. (2016). From trust in automation to decision neuroscience: Applying cognitive neuroscience methods to understand and improve interaction decisions involved in human automation interaction. *Frontiers in Human Neuroscience*, *10*, 290. https://doi.org/10.3389/fnhum.2016.00290

Ebel, P., Lingenfelder, C., & Vogelsang, A. (2023). Multitasking while driving: How drivers self-regulate their interaction with in-vehicle touchscreens in automated driving. *International Journal of Human–Computer Interaction*, *39*(16), 18. 1 https://doi.org/10.1080/10447318.2023.2215634

Edwards, J., Janssen, C., Gould, S., & Cowan, B. R. (2021). *Eliciting spoken interruptions to inform proactive speech agent design* [Paper presentation]. In Proceedings of the 3rd Conference on Conversational User Interfaces (Bilbao (Online), Spain) (CUI '21),. Association for Computing Machinery, New York, NY, Article 23, 12 pp. https://doi.org/10.1145/3469595.3469618

Feit, A. M., Weir, D., & Oulasvirta, A. (2016). How we type: Movement strategies and performance in everyday typing. In *Proceedings of the 2016 Chi Conference on Human Factors in Computing Systems*. ACM, 4262–4273.

Felten, F., Alegre, L. N., Nowe, A., Bazzan, A., Talbi, E. G., Danoy, G., & da Silva, B. C. (2023). *A toolkit for reliable benchmarking and research in multi-objective reinforcement learning* [Paper presentation]. Advances in Neural Information Processing Systems 36, (2023), 23671–23700.

Ferreira, C., Geig, M. (2018). Get started with the unity* entity component system (ECS), C# Job system, and burst compiler. Intel Corporation,[Online]. Available: https://www.intel.com/content/www/us/en/developer/articles/guide/get-started-with-the-unity-entity-component-system-ecs-c-sharp-job-system-and-burst-compiler.html *(visited on 2022-11-12)* 1 (2018), 1.

Fischer, R., & Plessow, F. (2015). Efficient multitasking: Parallel versus serial processing of multiple tasks. *Frontiers in Psychology*, *6*, 1366. https://doi.org/10.3389/fpsyg.2015.01366

Franke, T., Attig, C., & Wessel, D. (2019). A personal resource for technology interaction: Development and validation of the affinity for technology interaction (ATI) scale. *International Journal of Human–Computer Interaction*, *35*(6), 456–467. https://doi.org/10.1080/10447318.2018.1456150

Gasse, A., Lingler, A., Lorenz, M., Oulasvirta, A., Wintersberger, P., & Ebel, P. (2025). *Evaluating attention management systems for dynamic monitoring tasks* [Paper presentation]. In Adjunct Proceedings of the 4th Annual Symposium on Human-Computer Interaction for Work (CHIWORK '25 Adjunct),. Association for Computing Machinery, New York, NY, Article 14, 6 pp. https://doi.org/10.1145/3707640.3731920

Gebhardt, C., Hecox, B., van Opheusden, B., Wigdor, D., Hillis, J., Hilliges, O., & Benko, H. (2019). *Learning cooperative personalized policies from gaze data* [Paper presentation]. In Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (New Orleans, LA) (Uist '19)., Association for Computing Machinery, New York, NY, 197–208. https://doi.org/10.1145/3332165.3347933

Gebhardt, C., Oulasvirta, A., & Hilliges, O. (2020). Hierarchical reinforcement learning as a model of human task interleaving. *Computational Brain & Behavior*, *4*, 284–304. https://doi.org/10.1007/s42113-020-00093-9

Gheibi, O., & Weyns, D. (2022). Lifelong self-adaptation: Self-adaptation meets lifelong machine learning. In *Proceedings of the 17th Symposium on Software Engineering for Adaptive and Self-Managing Systems*. ACM, 1–12.

Gillie, T., & Broadbent, D. (1989). What makes interruptions disruptive? A study of length, similarity, and complexity. *Psychological Research*, *50*(4), 243–250. https://doi.org/10.1007/BF00309260

Guiard, Y., & Rioul, O. (2015). *A mathematical description of the speed/accuracy trade-off of aimed movement* [Paper presentation]. In Proceedings of the 2015 British HCI Conference (Lincoln, Lincolnshire, United Kingdom) (British HCI '15)., Association for Computing Machinery, New York, NY, 91–100. https://doi.org/10.1145/2783446.2783574

Hart, S. G., & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. A. Hancock & N. Meshkati (Eds.). *Human mental workload* Advances in Psychology., Vol. 52. 139–183. https://doi.org/10.1016/S0166-4115(08)62386-9

Hayes, C. F., Rădulescu, R., Bargiacchi, E., Källström, J., Macfarlane, M., Reymond, M., Verstraeten, T., Zintgraf, L. M., Dazeley, R., Heintz, F., et al. (2022). A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, *36*(1), 59. https://doi.org/10.1007/s10458-022-09552-y

Ho, J., & Intille, S. S. (2005). Using context-aware computing to reduce the perceived burden of interruptions from mobile devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 909–918.

Horvitz, E. (1999). *Principles of mixed-initiative user interfaces* [Paper presentation]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Pittsburgh, Pennsylvania) (CHI '99)., ACM, New York, NY, 159–166. https://doi.org/10.1145/302979.303030

Horvitz, E., Kadie, C., Paek, T., & Hovel, D. (2003). Models of attention in computing and communication: From principles to applications. *Communications of the ACM*, *46*(3), 52–59. https://doi.org/10.1145/636772.636798

Howes, A., Jokinen, J. P. P., & Oulasvirta, A. (2023). Towards machines that understand people. *AI Magazine*, *44*(3), 312–327. https://doi.org/10.1002/aaai.12116

Hudson, S., Fogarty, J., Atkeson, C., Avrahami, D., Forlizzi, J., Kiesler, S., Lee, J., & Yang, J. (2003). Predicting human interruptibility with sensors: A Wizard of Oz feasibility study. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 257–264.

Iqbal, HSarker. (2018). SilentPhone: Inferring user unavailability based opportune moments to minimize call interruptions. *EAI Endorsed Transactions on Mobile Communications and Applications*, *4*(15), e1. https://doi.org/10.4108/eai.10-7-2018.155647

Iqbal, S. T., & Bailey, B. P. (2005). *Investigating the effectiveness of mental workload as a predictor of opportune moments for interruption* [Paper presentation]. In CHI '05 Extended Abstracts on Human Factors in Computing Systems (Portland, OR) (Chi EA '05),. ACM, New York, NY, 1489–1492. https://doi.org/10.1145/1056808.1056948

Iqbal, S. T., & Bailey, B. P. (2010). A framework for linking notification delivery to the perceptual structure of goal-directed tasks. *ACM Transactions on Computer-Human Interaction (TOCHI)*, *17*(4), 1–28. https://doi.org/10.1145/1879831.1879833

Iqbal, S. T., & Horvitz, E. (2007). Disruption and recovery of computing tasks: Field study, analysis, and directions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 677–686.

Jokinen, J. P. P., Kujala, T., & Oulasvirta, A. (2021b). Multitasking in driving as optimal adaptation under uncertainty. *Human Factors*, *63*(8), 1324–1341. https://doi.org/10.1177/0018720820927687

Jokinen, J., Acharya, A., Uzair, M., Jiang, X., & Oulasvirta, A. (2021a). *Touchscreen typing as optimal supervisory control* [Paper presentation]. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (Yokohama, Japan) (Chi '21)., Association for Computing Machinery, New York, NY, Article 720, 14 pp. https://doi.org/10.1145/3411764.3445483

Jokinen, J., Ebel, P., Kujala, T. (2025). Predicting multitasking in manual and automated driving with optimal supervisory control. arXiv:2503.17993 [cs.HC] https://arxiv.org/abs/2503.17993

Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H., & Mattar, M. (2018). Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627* 1 (2018), 28.

Katidioti, I., Borst, J. P., & Taatgen, N. A. (2014). What happens when we switch tasks: Pupil dilation in multitasking. *Journal of Experimental Psychology: Applied*, *20*(4), 380. https://doi.org/10.1037/xap0000031

Katidioti, I., Borst, J. P., van Vugt, M. K., & Taatgen, N. A. (2016). Interrupt me: External interruptions are less disruptive than self-interruptions. *Computers in Human Behavior*, *63*, 906–915. https://doi.org/10.1016/j.chb.2016.06.037

Kaur, H., Williams, A. C., McDuff, D., Czerwinski, M., Teevan, J., & Iqbal, S. T. (2020). *Optimizing for happiness and productivity: Modeling opportune moments for transitions and breaks at work* [Paper presentation]. In Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (Honolulu, HI) (Chi '20)., Association for Computing Machinery, New York, NY, 1–15. https://doi.org/10.1145/3313831.3376817

Kim, A., Choi, W., Park, J., Kim, K., & Lee, U. (2018). Interrupting drivers for interactions: Predicting opportune moments for in-vehicle proactive auditory-verbal tasks. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 2*, 4 (2018), 1–28.

Knowles, K., Witbrock, M., Dobbie, G., & Yogarajan, V. (2023). A proposal for a language model based cognitive architecture. In *Proceedings of the AAAI Symposium Series*, Vol. 2. AAAI Press, 295–301. https://doi.org/10.1609/aaaiss.v2i1.27691

Kool, W., McGuire, J. T., Rosen, Z. B., & Botvinick, M. M. (2010). Decision making and the avoidance of cognitive demand. *Journal of Experimental Psychology: General*, *139*(4), 665. https://doi.org/10.1037/a0020198

Liao, P., Greenewald, K., Klasnja, P., & Murphy, S. (2020). Personalized heartsteps: A reinforcement learning algorithm for optimizing physical activity. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–22. https://doi.org/10.1145/3381007

Lingler, A., Frijns, H. A., Boess, L., Murziakova, N., & Wintersberger, P. (2026). *Using LLMs to model notification timing from context-sensitive features* [Paper presentation]. In Proceedings of the Augmented Humans International Conference 2026 (AHs '26)., Association for Computing Machinery, New York, NY, 406–419. https://doi.org/10.1145/3795011.3795067

Lingler, A., Talypova, D., & Wintersberger, P. (2024a). *AITentive: A Toolkit to develop RL-based attention management systems* [Paper presentation]. In Adjunct Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology (Pittsburgh, PA) (UIST Adjunct '24)., Association for Computing Machinery, New York, NY, Article 82, 3 pp. https://doi.org/10.1145/3672539.3686314

Lingler, A., Talypova, D., & Wintersberger, P. (2025). Evaluating the effect of different multitasking conditions in al-supported attention management systems. In *HHAI 2025: Proceedings of the 4th International Conference on Hybrid Human-Artificial Intelligence*. SAGE Publications 1 Oliver's Yard, 55 City Road, London, EC1Y 1SP, IOS Press, 385–400.

Lingler, A., Talypova, D., Jokinen, J. P. P., Oulasvirta, A., & Wintersberger, P. (2024b). Supporting task switching with reinforcement learning. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. ACM, 1–18.

Lorenz, M., Konzack, N., Lingler, A., Wintersberger, P., & Ebel, P. (2026). *CR-Eyes: A computational rational model of visual sampling behavior in atari games* [Paper presentation]. In Proceedings of the Extended Abstracts of the 2026 CHI Conference on Human Factors in Computing Systems (CHI EA '26)., Association for Computing Machinery, New York, NY, Article 153, 6 pp. https://doi.org/10.1145/3772363.3798498

Luo, Y., Lee, B., Kim, Y.-H., & Choe, E. K. (2022). NoteWordy: Investigating touch and speech input on smartphones for personal data capture. *Proc. ACM Hum.-Comput. Interact.* 6, ISS, Article 581 (Nov. 2022), 24 pp. https://doi.org/10.1145/3567734

MacKenzie, I. S., & Soukoreff, R. W. (2002). Text entry for mobile computing: Models and methods, theory and practice. *Human–Computer Interaction*, *17*(2-3), 147–198. https://doi.org/10.1207/S15327051HCI172&3_2

Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016a). *Resource management with deep reinforcement learning* [Paper presentation]. In Proceedings of the 15th ACM Workshop on Hot Topics in Networks (Atlanta, GA) (HotNets, '16). Association for Computing Machinery, New York, NY, 50–56. https://doi.org/10.1145/3005745.3005750

Mao, H., Alizadeh, M., Menache, I., & Kandula, S. (2016b). Resource management with deep reinforcement learning. In *Proceedings of the 15th ACM Workshop on Hot Topics in Networks*. ACM, 50–56.

Mao, H., Schwarzkopf, M., Venkatakrishnan, S. B., Meng, Z., & Alizadeh, M. (2019). Learning scheduling algorithms for data processing clusters. In *Proceedings of the ACM Special Interest Group on Data Communication*. ACM, 270–288.

Mark, G., Gonzalez, V. M., & Harris, J. (2005). No task left behind? Examining the nature of fragmented work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 321–330.

Mark, G., Gudith, D., & Klocke, U. (2008). The cost of interrupted work: More speed and stress. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 107–110.

Miazga, M. P., Bussmann, H., Oulasvirta, A., & Ebel, P. (2026). *Log2Motion: Biomechanical motion synthesis from touch logs* [Paper presentation]. In Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)., Association for Computing Machinery, New York, NY, Article 1038, 22 pp. https://doi.org/10.1145/3772318.3790773

Monk, C. A., Boehm-Davis, D. A., Mason, G., & Trafton, J. G. (2004). Recovering from interruptions: Implications for driver distraction research. *Human Factors*, *46*(4), 650–663. https://doi.org/10.1518/hfes.46.4.650.56816

Morgan, B., D'Mello, S., Abbott, R., Radvansky, G., Haass, M., & Tamplin, A. (2013). Individual differences in multitasking ability and adaptability. *Human Factors*, *55*(4), 776–788. https://doi.org/10.1177/0018720812470842

Okoshi, T., Ramos, J., Nozaki, H., Nakazawa, J., Dey, A. K., & Tokuda, H. (2015). *Reducing users' perceived mental effort due to interruptive notifications in multi-device mobile environments* [Paper presentation]. In Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing (Osaka, Japan) (UbiComp '15),. Association for Computing Machinery, New York, NY, 475–486. https://doi.org/10.1145/2750858.2807517

Oulasvirta, A., & Saariluoma, P. (2006). Surviving task interruptions: Investigating the implications of long-term working memory theory. *International Journal of Human-Computer Studies*, *64*(10), 941–961. https://doi.org/10.1016/j.ijhcs.2006.04.006

Oulasvirta, A., Jokinen, J. P. P., & Howes, A. (2022). *Computational rationality as a theory of interaction* [Paper presentation]. In Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (New Orleans, LA) (Chi '22)., Association for Computing Machinery, New York, NY, Article 359, 14 pp. https://doi.org/10.1145/3491102.3517739

Oulasvirta, A., Tamminen, S., Roto, V., & Kuorelahti, J. (2005). Interaction in 4-second bursts: The fragmented nature of attentional resources in mobile HCI. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 919–928.

Pejovic, V., & Musolesi, M. (2014). InterruptMe: Designing intelligent prompting mechanisms for pervasive applications. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. ACM, 897–908.

Pejovic, V., & Musolesi, M. (2015). Anticipatory mobile computing: A survey of the state of the art and research challenges. *ACM Computing Surveys (CSUR)*, *47*(3), 1–29. https://doi.org/10.1145/2693843

Pielot, M., Cardoso, B., Katevas, K., Serrà, J., Matic, A., & Oliver, N. (2017). Beyond interruptibility: Predicting opportune moments to engage mobile phone users. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 3 (2017), 1–25.

Romero, O. J., Zimmerman, J., Steinfeld, A., & Tomasic, A. (2023). Synergistic integration of large language models and cognitive architectures for robust ai: An exploratory analysis. In *Proceedings of the AAAI Symposium Series*, Vol. 2. AAAI Press, 396–405. https://doi.org/10.1609/aaaiss.v2i1.27706

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

Salvucci, D. D., & Bogunovich, P. (2010). *Multitasking and monotasking: The effects of mental workload on deferred task interruptions* [Paper presentation]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Atlanta, Georgia) (Chi '10),. Association for Computing Machinery, New York, NY, 85–88. https://doi.org/10.1145/1753326.1753340

Salvucci, D. D., Taatgen, N. A., & Borst, J. P. (2009). *Toward a unified theory of the multitasking continuum: From concurrent performance to task switching, interruption, and resumption* [Paper presentation]. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Boston, MA) (Chi '09),. Association for Computing Machinery, New York, NY, 1819–1828. https://doi.org/10.1145/1518701.1518981

Schneegass, C., & Draxler, F. (2021). Designing task resumption cues for interruptions in mobile learning scenarios. In E. Niforatos and T. Dingler (Eds.). *Technology-augmented perception and cognition* (pp. 125–181). Springer International Publishing. Series Title: Human–Computer Interaction Series. https://doi.org/10.1007/978-3-030-30457-7_5

Shi, D., Zhu, Y., Fernandes Junior, F. E., Zhai, S., & Oulasvirta, A. (2025). *Simulating errors in touchscreen typing* [Paper presentation]. In Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)., Association for Computing Machinery, New York, NY, Article 1086, 13 pp. https://doi.org/10.1145/3706598.3713153

Shi, D., Zhu, Y., Jokinen, J. P. P., Acharya, A., Putkonen, A., Zhai, S., & Oulasvirta, A. (2024). CRTypist: Simulating touchscreen typing behavior via computational rationality. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*. ACM, 1–17.

Speier, C., Valacich, J. S., & Vessey, I. (1999). The influence of task interruption on individual decision making: An information overload perspective. *Decision Sciences*, *30*(2), 337–360. https://doi.org/10.1111/j.1540-5915.1999.tb01613.x

Speier, C., Vessey, I., & Valacich, J. S. (2003). The effects of interruptions, task complexity, and information presentation on computer-supported decision-making performance. *Decision Sciences*, *34*(4), 771–797. https://doi.org/10.1111/j.1540-5414.2003.02292.x

Srivastava, N., Jain, R., Healey, J., Bylinskii, Z., & Dingler, T. (2021). *Mitigating the Effects of Reading Interruptions by Providing Reviews and Previews* [Paper presentation]. In Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems (Yokohama, Japan) (Chi EA '21)., Association for Computing Machinery, New York, NY, Article 229, 6 pp. https://doi.org/10.1145/3411763.3451610

Steil, J., Müller, P., Sugano, Y., & Bulling, A. (2018). Forecasting user attention during everyday mobile interactions using device-integrated and wearable sensors. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services*. ACM, 1–13.

Stisen, A., Blunck, H., Kjærgaard, M. B., Prentow, T. S., Mathisen, A., Sørensen, S., & Grønbæk, K. (2017). Task phase recognition and task progress estimation for highly mobile workers in large building complexes. *Pervasive and Mobile Computing*, *38*, 418–429. https://doi.org/10.1016/j.pmcj.2016.08.016

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT Press.

Talypova, D., Lingler, A., & Wintersberger, P. (2023). User-centered investigation of features for attention management systems in an online vignette study. In *Proceedings of the 22nd International Conference on Mobile and Ubiquitous Multimedia*. ACM, 108–121.

Talypova, D., Vesic, A., Shahu, A., Frijns, H. A., & Wintersberger, P. (2026). *Decomposing autonomy: Explaining AI technology acceptance through a liberty-based framework* [Paper presentation]. In Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)., Association for Computing Machinery, New York, NY, Article 1044, 17 pp. https://doi.org/10.1145/3772318.3791789

Thil, L.-A., Popa, M., & Spanakis, G. (2024). Navigating webai: Training agents to complete web tasks with large language models and reinforcement learning. In *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*. ACM, 866–874.

Trafton, J. G., Altmann, E. M., Brock, D. P., & Mintz, F. E. (2003). Preparing to resume an interrupted task: Effects of prospective goal encoding and retrospective rehearsal. *International Journal of Human-Computer Studies*, *58*(5), 583–603. https://doi.org/10.1016/S1071-5819(03)00023-5

Wintersberger, P., Riener, A., Schartmüller, C., Frison, A.-K., & Weigl, K. (2018). *Let me finish before I take over: Towards attention aware device integration in highly automated vehicles* [Paper presentation]. In Proceedings of the 10th International Conference on Automotive User Interfaces and Interactive Vehicular Applications (Toronto, ON, Canada) (AutomotiveUI '18)., Association for Computing Machinery, New York, NY, 53–65. https://doi.org/10.1145/3239060.3239085

Wintersberger, P., Schartmüller, C., & Riener, A. (2019). Attentive user interfaces to improve multitasking and take-over performance in automated driving: The auto-net of things. *International Journal of Mobile Human Computer Interaction (IJMHCI)*, *11*(3), 40–58. https://doi.org/10.4018/IJMHCI.2019070103

Wylie, G., & Allport, A. (2000). Task switching and the measurement of "switch costs". *Psychological Research*, *63*(3), 212–233. https://doi.org/10.1007/s004269900003

Xu, S., & Zhang, X. (2023). *Augmenting human cognition with an AI-mediated intelligent visual feedback* [Paper presentation]. In Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (Hamburg, Germany) (Chi '23)., Association for Computing Machinery, New York, NY, Article 26, 16 pp. https://doi.org/10.1145/3544548.3580905

Yang, N., Chen, S., Zhang, H., & Berry, R. (2025a). Beyond the edge: An advanced exploration of reinforcement learning for mobile edge computing, its applications, and future research trajectories. *IEEE Communications Surveys & Tutorials*, *27*(1), 546–594. https://doi.org/10.1109/COMST.2024.3405075

Yang, N., Wen, J., Zhang, M., & Tang, M. (2025b). Generalizable pareto-optimal offloading with reinforcement learning in mobile edge computing. *IEEE Transactions on Services Computing*, *18*(6), 3824–3836. https://doi.org/10.1109/TSC.2025.3604371

Yu, D., Desai, R., Zhang, T., Benko, H., Jonker, T. R., & Gupta, A. (2022). *Optimizing the timing of intelligent suggestion in virtual reality* [Paper presentation]. In Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology (Bend, or, USA) (UIST '22)., Association for Computing Machinery, New York, NY, Article 6, 20 pp. https://doi.org/10.1145/3526113.3545632

Zhao, W., Queralta, J. P., & Westerlund, T. (2020). *Sim-to-real transfer in deep reinforcement learning for robotics: A survey* [Paper presentation]. In 2020 IEEE Symposium Series on Computational Intelligence (SSCI)., IEEE, Piscataway, NJ, 737–744. https://doi.org/10.1109/SSCI47803.2020.9308468

Züger, M., Müller, S. C., Meyer, A. N., & Fritz, T. (2018). *Sensing interruptibility in the office: A field study on the use of biometric and computer interaction sensors* [Paper presentation]. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (Montreal, QC, Canada) (CHI '18)., Association for Computing Machinery, New York, NY, 1–14. https://doi.org/10.1145/3173574.3174165

## About the authors

**Alexander Lingler**: His research models human cognition to develop domain- and task-independent attention management systems. His PhD explores adaptive methods for improving interaction and multitasking, including work on the FWF project AITentive. With a computer science degree from TU Wien, he combines cognitive science, adaptive systems, and HCI to advance user interfaces.

**Helena Anna Frijns**: Her research focuses on Human-Robot Interaction, especially modeling communication between people and robots and designing interactions that help users interpret and adapt robot behavior. As a researcher in the Intelligent User Interfaces group with a doctorate from TU Wien, she explores adaptation, mental models, and interaction design for collaborative robotics.

**Nelson Silva**: He is a senior lecturer and researcher, leading the IT:U Extended Reality and Interaction Labs. His research focuses on eye-tracking support techniques and adaptive user interfaces, including XR interfaces and AI-supported visual analytics systems for Digital Twins. His work addresses sensing user behavior and inferring interests to support analytical tasks.

**Patrick Ebel**: He is an Assistant Professor for Computational Interaction at Hasso Plattner Institute, where he leads the CIAO research group. His research focuses on simulated users and computational models of human cognition, perception, and motor control using data-driven methods and reinforcement learning. Previously, he led a Junior Research Group at ScaDS.AI.

**Philipp Wintersberger**: He is Professor of Intelligent User Interfaces at IT:U. He leads an interdisciplinary team of scientists working on projects funded by FWF, the Austrian Research Promotion Agency (FFG), and industry partners. His work focuses on human-machine cooperation in safety-critical environments, including automated vehicles, robotic and industrial systems, and AI-supported workplaces.