PrISM-Observer: Intervention Agent to Help Users Perform Everyday Procedures Sensed using a Smartwatch

Riku Arakawa
Carnegie Mellon University
Pittsburgh, USA
rarakawa@cs.cmu.edu

Hiromu Yakura
Max-Planck Institute for Human Development
Berlin, Germany
yakura@mpib-berlin.mpg.de

Mayank Goel
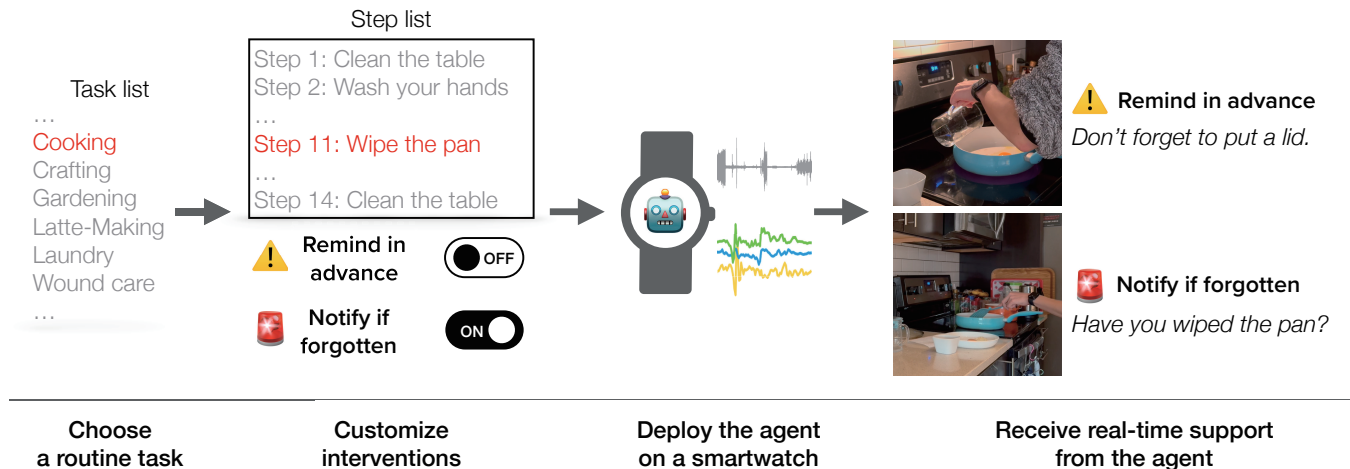Carnegie Mellon University
Pittsburgh, USA
mayankgoel@cmu.edu

Figure 1: PrISM-Observer is a framework to design and trigger interventions to mitigate user errors in daily procedural tasks such as cooking. The user actions are sensed using a smartwatch, which can be used to deliver the interventions and integrate into the user's life to observe their task execution. The system combines multimodal sensing (sound + motion) with stochastic modeling of user actions to forecast the intervention moment. It reminds users of key steps at the optimal moment and notifies them if they forget a step in real-time.

ABSTRACT

We routinely perform procedures (such as cooking) that include a set of atomic steps. Often, inadvertent omission or misordering of a single step can lead to serious consequences, especially for those experiencing cognitive challenges such as dementia. This paper introduces *PrISM-Observer*, a smartwatch-based, context-aware, real-time intervention system designed to support daily tasks by preventing errors. Unlike traditional systems that require users to seek out information, the agent observes user actions and intervenes proactively. This capability is enabled by the agent's ability to continuously update its belief in the user's behavior in real-time through multimodal sensing and forecast optimal intervention moments and methods. We first validated the steps-tracking performance of our framework through evaluations across three datasets with different complexities. Then, we implemented a real-time agent system using a smartwatch and conducted a user study in a cooking task scenario. The system generated helpful interventions, and we gained positive feedback from the participants. The general applicability of PrISM-Observer to daily tasks promises broad applications, for instance, including support for users requiring more involved interventions, such as people with dementia or post-surgical patients.

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**; *Interactive systems and tools*.

KEYWORDS

context-aware intervention, procedure tracking, task assistant

1 INTRODUCTION

Every day, we perform many tasks, ranging from cooking to crafting to self-care, which involve a series of atomic steps. Accurately executing all the steps of tasks can be difficult, especially when the tasks become routine and fail to capture our full attention [21] or when we face cognitive challenges such as dementia [25]. For example, people often forget to turn on the washing machine after loading it or turn off lights before leaving home [1]. Such mistakes where we omit essential steps or confuse the order of actions can lead to undesirable outcomes [41]. For instance, in a study with over a hundred participants, close to 20% of participants made critical errors while using COVID-19 self-test kits [39]. Thus, real-time assistance by sensing a user's actions and intervening as needed can help improve quality of life.

Most of the existing task-support solutions in HCI are tailored for specific activities, often necessitating specialized equipment or advanced computers equipped with cameras and displays, such as Augmented Reality (AR) glasses. For instance, Uriu *et al.* [50] developed a sensor-equipped frying pan that offers context-sensitive information like the pan's current temperature. Also, *AdapTutAR* [22] is a machine task tutoring system designed to monitor users in following tutorials and offer feedback through AR glasses. However, we often need help with mundane tasks or in situations where instrumenting ourselves or the environment is not easy. There is a need for a solution that seamlessly integrates into supporting various routine tasks and is practical for constant use. Additionally, current systems predominantly rely on users actively seeking information, like consulting a recipe while cooking [43, 50]. There has been limited exploration into passive interactions, where systems proactively monitor and offer corrective feedback when errors occur. Designing such interactions is complex, as the system needs to model the user's spontaneous behavior to predict errors and intervene without becoming intrusive or annoying. The challenge intensifies when we shift focus from camera-based methods to more practical and ubiquitous methods such as motion and sound sensors on smartwatches. These solutions offer seamless assimilation into daily life while lowering privacy concerns but come at the cost of sensor accuracies. Our prior work [8] tried addressing the errors of sound and motion-based Human Activity Recognition (HAR) models by combining procedure knowledge with multimodal sensor data. However, we have yet to use these advances to build a reliable intervention system that guides a user through a variety of tasks in real-time.

To achieve an agent system that intervenes to mitigate errors in daily procedural tasks, this paper introduces *PrISM-Observer*. It preemptively models user task behavior and optimizes intervention methods and timing by considering uncertainties for the sensing data and anticipating the user's future behavior. Moreover, the framework allows the design of user-friendly reminder-based interventions that the user or a system designer can customize. Using the framework, we developed a prototype system that operates on a smartwatch (Figure 1) – a device chosen for its ubiquity, minimal privacy concerns compared to camera-based systems, and capability to monitor a user across various daily activities. The prototype offers timely and relevant interventions with minimal reliance on task-specific rules.

PrISM-Observer either reminds users to execute a step in advance (REMIND IN ADVANCE), or if it infers that the user may have forgotten a step, it notifies them separately (NOTIFY IF FORGOTTEN). These interventions are time-critical as a reminder that comes too early or too late will be useless. To verify PrISM-Observer's ability to optimize the intervention timing, we applied the proposed framework to three tasks with different procedural complexities: wound care, cooking, and latte-making (Study 1). The results showed the proposed approach reduced the timing error compared to a baseline approach that does not use sensor information: by averaging across all steps, 26.8 (baseline) → 24.1 (proposed) seconds in the wound care, 119.5 → 61.5 seconds in the cooking, and 50.1 → 22.3 seconds in the latte-making tasks, respectively.

Subsequently, we built a real-time agent system that assisted the users in a specific cooking task of making a sunny-side-up and a grilled sausage. We evaluated the agent through a user study (Study 2, $N = 10$) to examine the system's performance and usability. The results showed that the participants perceived the triggered interventions to be accurate (20 out of the total 27 triggered interventions were scored five or higher for accuracy on the 7-point Likert scale in the post-task questionnaire). We also found a trend between the delay and perceived accuracy for the two kinds of intervention, highlighting the importance of optimizing their timing. In addition, the NOTIFY IF FORGOTTEN interventions were triggered correctly (22 out of the total 25 intervention chances), verifying the intervention policy's effectiveness. Overall, the participants found the system reliable and showed a positive behavioral intention (8 out of 10 participants). Furthermore, their comments validated our design of using different types of interventions and making them customizable, as well as offered implications for the interaction between humans and real-world task-support agents.

In this paper, we make the following contributions:

(1) framework for modeling user behavior in procedural tasks and designing interventions, building upon the foundation of the existing multimodal procedure tracking module [8].
(2) comprehensive evaluation across three daily-task datasets demonstrated the proposed approach's superiority in forecasting the optimal moments for interventions, highlighting its effectiveness in complex tasks with over 50% timing error reduction.
(3) user study with a real-time prototype system on a smartwatch, which not only showed its preferable experience but also informed design implications to build reliable and acceptable task-support agents.

It is important to note that PrISM-Observer performs promisingly, even though the underlying HAR models are approximately only 50% accurate at detecting each atomic step of the procedures. This result demonstrates that it is possible to use imperfect sensing and machine learning to build a useful intervention system to aid an imperfect human prone to making mistakes. Our system will be particularly helpful for users who face cognitive challenges (*e.g.*, patients with dementia or after surgery), and we are working closely with such populations as part of our future work. We open-source the framework and the dataset to facilitate the research in this domain (https://github.com/cmusmashlab/prism).

2 BACKGROUND AND RELATED WORK

We first examine the cognitive psychology literature on human errors in daily tasks, recognizing that such errors are inevitable and necessitate support. Next, we review HCI studies concerning assistants for procedural tasks, underscoring the importance of using a prevalent device to support everyday procedures and its unique challenges. Finally, we explore multimodal sensing research related to activity recognition, especially using a smartwatch.

2.1 Human Errors in Everyday Tasks

The improper execution of everyday tasks, ranging from cooking and medical self-care to machine use, is a multifaceted issue rooted in cognitive psychology. Studies have shown that errors are likely to occur when the working memory load is high or the user is not fully attentive [13, 17]. For instance, multitasking during cooking can lead to oversights or mistakes in recipe execution [21]. Also, Beaver *et al.* [10] discussed that complex and integrative steps in daily activities may be the first to be affected by cognitive decline. In fact, cognitive challenges such as memory lapses contribute significantly to non-adherence to self-care activities by the elderly [36]. Additionally, misunderstanding instructions can easily lead to critical errors; almost 20% of participants made mistakes while using COVID-19 self-test kits [39]. These examples underscore the human error-prone nature in various daily scenes [41], highlighting the need for situated support that helps users avoid or recover from errors to compensate for cognitive limitations, as emphasized by Zhang [56]

2.2 Assistants for Procedural Tasks

Supporting users in conducting complicated tasks in the real world has been a popular research theme in HCI research. A common approach involves crafting specialized devices tailored for particular activities or tasks. For example, Lee and Dey [29] developed a sensor-augmented pillbox and feedback system to improve medication compliance. In the cooking domain, *Cooking Navi* [19] is an interface providing multimedia recipe information (*i.e.*, text, video, and audio) to aid in cooking. Uriu *et al.* [50] extended this support system by creating a sensor-equipped frying pan that offers context-sensitive information like the pan's current temperature. Similarly, *MimiCook* [43] combines a depth camera and projector to deliver on-the-spot guidance during cooking.

Computer-vision-based approaches are popular to guide users through various tasks [20, 22, 32, 47, 55]. For instance, *AR Cooking* [20] used 3D animation of cookware on AR glasses. Serván *et al.* [47] developed a system to overlay work instruction in an assembly task using AR. Similarly, *AdapTutAR* [22] is a machine task tutoring system that monitors learners' tutorial-following status and provides feedback via AR glasses. *HoloAssist* [53] is a system where a human observer watches the task performer's egocentric video captured by AR glasses and guides them verbally. To support the creation of such technologies, researchers have compiled datasets capturing first-person perspectives on procedural tasks such as cooking [38] or assembly [44, 46].

Despite the success of these systems, using cameras and displays can result in privacy-invasive and power-hungry systems. This issue becomes apparent when we want to support users' daily routines pervasively in contrast to specific, high-stakes situations such as assembly [42]. Furthermore, these solutions predominantly offer context-aware information, assuming that users actively seek the information. While benefiting those inexperienced with the task, these systems may not assist adept users who might still commit errors due to inattentiveness or cognitive overload. A similar motivation for error-checking systems was discussed by Bovo *et al.* [12], who proposed an approach for real-time error prediction in sequence-constrained procedural tasks. While showing promising performance in detecting errors in an item-picking-placement task, the method relies on the assumption of a predefined step sequence and task-specific heuristics like item location, which does not apply to various tasks where users are permitted to exhibit multiple behavioral patterns.

Hence, this paper aims to create a generalizable framework to monitor and intervene with users engaged in everyday tasks. Given such circumstances, users might not always prefer to utilize sophisticated equipment like AR glasses or external cameras for every task. Consequently, given its widespread use and minimal interference with the task, we have chosen a smartwatch as our device. This decision undoubtedly introduces a research question: *how can we design a reliable intervention agent system for users' spontaneous behavior using imperfect sensing?* Our solution is to stochastically model user behavior from sensor observation with transition knowledge to trigger situated interventions. We use minimized task-specific assumptions to offer flexibility and cater to different user preferences.

2.3 Smartwatch-Based Context Sensing

Human Activity Recognition (HAR) is a widely studied technology that senses user actions and behavior [54]. Among many modalities used for HAR, smartwatch-based systems often use audio and motion data [11, 18, 26, 27, 34]. For example, *ViBand* [27] enabled bio-acoustic sensing using commodity smartwatch accelerometers, detecting the use of different hand-held tools such as a toothbrush and heat gun. Ashry *et al.* [9] used cascading bidirectional long short-term memory to classify motion data of different daily activities. For audio, *Ubicoustics* [27] employed acoustic sensing to classify 30 daily activities, such as hand washing and typing. Recent work has proposed a multimodal learning approach to maximize the capability, for instance, *GestEar* [11], which combined audio and motion signals to distinguish different gestures, such as knocking and snapping. Moreover, *SAMoSA* [34] explored the downsampling of the audio modality by supplementarily using the motion data to make the system more privacy-aware. These advances in the smartwatch's HAR capability led to deployed applications, such as Apple Watch's Handwash detection feature [5].

We aim to create an intelligent agent that assists users in preventing errors in various tasks. Here, despite its potential, a significant challenge with HAR is its limited accuracy when extended to a wide range of real-world activities. For instance, Liaqat *et al.* [31] highlighted the difficulties posed by noise in tasks performed outside controlled environments. Additionally, when HAR is employed for tracking procedural tasks, distinguishing between steps with similar signal profiles is often hard. In this regard, studies like those by Nakauchi *et al.* [35] and Arakawa *et al.* [8] have suggested the

potential of enhancing tracking accuracy by using transition information between steps of a task. While these approaches have improved the accuracy of procedure tracking, the performance is still far from perfect. Thus, a fully functional interactive system for procedural task support on a smartwatch has yet to be realized. This work provides an approach to using state-of-the-art activity recognition and procedure tracking algorithms to design a reliable agent system.

3 FRAMEWORK FOR ERROR MONITORING IN PROCEDURAL TASKS

In this section, we introduce a framework for monitoring users' actions and offering timely interventions. We first outline a user scenario to illustrate the utility of such an agent system and introduce our intervention design. Then, we present an algorithm using multimodal sensing to forecast user actions and a policy to trigger interventions.

3.1 User Scenario

As discussed in Section 2.1, a user's performance in daily tasks is affected by various factors, such as inattention and cognitive load or time pressure. Consider two scenarios: *1) Tom decided to prepare a sunny-side-up for breakfast instead of his usual choice of scrambled egg. Given he is not used to this task, he inadvertently forgets to add oil to the pan before cracking an egg. Consequently, the egg sticks to the surface of the pan. When Tom attempts to lift it with a spatula, the yolk breaks, leading to a disappointing start to the day. 2) Catherine is preparing a latte using a semi-automatic coffee machine in her office's shared kitchen. She is in a hurry to get to her next meeting and forgets to clean the steam wand after use. As a result, milk residue clogs the holes of the wand, necessitating maintenance.* In these situations, it would be helpful if a system could remind users just in time to avoid errors or notify them when errors are detected in real-time. We envision a context-aware agent living on the user's watch that works in the following way: *1) Just as Tom is about to crack the egg, he receives a reminder on the watch to pour oil into the pan. 2) When Catherine is about to leave the kitchen with her latte, her watch notifies her to clean the steam wand.* The following subsections describe the formulation and implementation to achieve such monitoring and intervention.

3.2 Interaction Design

To enable support for a diverse set of tasks, we design our interactions in a generalizable manner instead of using task-specific heuristics as discussed in Section 2.2. Let $S = \{s_1, s_2, \ldots s_N\}$ be the set of the atomic steps in the procedure where N is the total number of the steps. We assume a transition graph G, which contains information on the average time spent on each step and the transition probabilities between steps. G can be obtained from sample demonstrations of the task as described by Arakawa *et al.* [8].

Current research on procedure tracking treats each step of a process equally, focusing on accuracy in recognizing each step on a frame-by-frame basis [8, 35]. In contrast, to develop a user-centered helpful agent, it is crucial to prioritize the steps each user needs support for. Considering Tom's cooking example above, it might not be necessary to verify whether he remembered to bring an egg
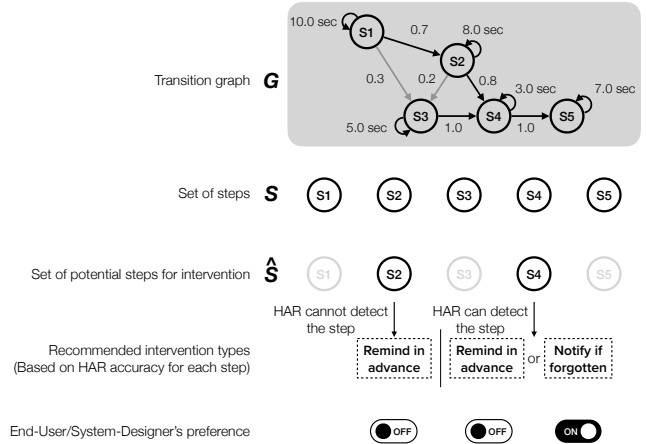

Figure 2: Process of selecting steps for intervention and their types. The system suggests the possibility of the NOTIFY IF FORGOTTEN intervention based on the HAR accuracy of the step. Refer to Section 3.2 for details.

from the refrigerator, as this action inevitably happens during the cooking process. Therefore, we assume a subset $\hat{S} \in S$ as a set of steps for which situated interventions can be helpful.

Given the nature of each step in $\hat{S}$, suitable interventions may change. For example, users may want to receive notifications only when they forget a specific step, *i.e.*, error detection. Or, users might appreciate receiving preemptive reminders for crucial steps, especially when the timing and sequence are critical and irreversible – for instance, adding oil to the pan before cracking an egg into it. Providing users with a global control like this is key to successfully developing human-AI interaction systems [2]. Accordingly, we prepared two types of interventions that will be assigned to each target step $\hat{s} \in \hat{S}$: REMIND IN ADVANCE and NOTIFY IF FORGOTTEN. The REMIND IN ADVANCE intervention is intended to happen before the user starts the step. On the other hand, the NOTIFY IF FORGOTTEN intervention only happens when the user forgets the step. To reliably achieve this notification (*i.e.*, preventing false positives or negatives), the system must be able to detect whenever the step happens accurately so that, when the step is not inferred, the system can be confident that the user missed the step. Thus, we enabled the system to suggest the possibility of using the NOTIFY IF FORGOTTEN intervention for a step based on the step's detectability from sensors, the detail of which is described in Section 3.3.2, hence satisfying a key requirement for successful human-AI interaction – communicating how well the system functions [2]. In addition, PrISM-Observer's default intervention messages support users' efficient dismissal in case of false predictions without annoying them [2]. For the REMIND IN ADVANCE intervention, the message is "Don't forget to do $\hat{s}$," while it is "Have you done $\hat{s}$?" for the NOTIFY IF FORGOTTEN intervention.

Lastly, system designers or end-users can finalize the configuration, *i.e.*, which $\hat{s} \in \hat{S}$ are supported by interventions and the intervention type. For instance, a kitchen manager in Catherine's office may be particularly interested in ensuring that cleaning occurs

post-use of the machinery, thus enabling the NOTIFY IF FORGOTTEN intervention for the step. Figure 2 summarizes the entire selection process for intervention.

3.3 Intervention Timing Optimization

In our framework, the system persistently observes user behavior to provide situated interventions, triggering them at opportune times. A significant obstacle arises from the fact that the training data often lacks instances of actual user errors, and it is hard to train a model to predict or detect errors directly. Thus, identifying omitted actions or reminding before a specific step in real-time is particularly challenging, especially when steps can be completed in various sequences, unlike prior work assuming a constrained sequence [12]. To address this, we propose an alternative strategy to forecast when a specific step, $\hat{s}$, should occur based on the current belief about the user state – preemptively estimating the remaining time till $\hat{s}$ happens. We refer to this remaining time till the user reaches $\hat{s}$ from the current step (at time t) by using a stochastic variable $D_t^{\hat{s}}$, which we describe in detail later in the next subsection. At a high level, this framework decides whether to trigger an intervention based on the expectation of when the user will perform the step $\hat{s}$. Once the certainty of the timing of a step goes above an empirically determined threshold, PrISM-Observer prepares a *timer* to trigger an intervention corresponding to the step. This timer functions like this: if the system estimates the user will be doing a certain step in 10 seconds with high confidence, it waits for *slightly less than* 10 seconds to trigger the REMIND IN ADVANCE intervention, or the NOTIFY IF FORGOTTEN if the user does not do the step around those *slightly more than* 10 seconds.

Algorithm 1 presents the overview of the framework. In this pseudo-code, for simplicity, we assume a single target step $\hat{s}$, but the framework is extendable to multiple target steps in parallel, as demonstrated in the user study later. In the following subsections, we discuss the modeling of $D_t^{\hat{s}}$ and intervention policy.

3.3.1 Stochastic Modeling of User Behavior.
We use a stochastic variable $D_t^{\hat{s}}$, indicating a remaining time at the given t till the user reaches the target step $\hat{s}$. $D_t^{\hat{s}}$ follows a probabilistic distribution $P(D_t^{\hat{s}})$. The expectation can be calculated as,

$$E[D_t^{\hat{s}}] = \sum D_t^{\hat{s}} P(D_t^{\hat{s}})$$
$$= \sum_{s \in S} P(s) \sum_{\tau \in T_s^{\hat{s}}} P(\tau) \text{Time}(s \to \hat{s}, \tau)$$

, where $T_s^{\hat{s}}$ is the set of the possible trajectories from step s to the target step $\hat{s}$, $P(s)$ indicates the probability of the user being at step s at time t, $P(\tau)$ indicates the probability of the user will follow the trajectory τ, and $\text{Time}(s \to \hat{s}, \tau)$ means the average time it takes to transition from step s to $\hat{s}$ by following the path τ. $P(s)$ is obtained by PrISM-Tracker [8]'s output while $P(\tau)$ and $\text{Time}(s \to \hat{s}, \tau)$ are calculated based on the transition graph G.

Importantly, this equation tells us two uncertainties with respect to user behavior: uncertainty in the current state (first sigma term) and uncertainty in the future trajectory (second sigma term). To

---

Algorithm 1 PrISM-Observer's algorithmic framework

1: **User Input:** $\hat{s}$ and its intervention type
2: **System Configuration:** trained *PrISM-Tracker* [8] (including transition graph G), threshold $h^{\hat{s}}$, offset constants K^+ and K^-
3: begin the task
4: **while** the user has not finished the task **do**
5:     get the latest frame sensor data
6:     apply PrISM-Tracker and update the internal state
7:     calculate $E[D_t^{\hat{s}}]$ and $H[D_t^{\hat{s}}]$ (Section 3.3.1)
8:     **if** intervention timer should begin (Section 3.3.2) **then**
9:         **if** intervention is REMIND IN ADVANCE **then**
10:             timer $\leftarrow E[D_t^{\hat{s}}] - K^-$
11:         **else if** intervention is NOTIFY IF FORGOTTEN **then**
12:             timer $\leftarrow E[D_t^{\hat{s}}] + K^+$
13:         **end if**
14:         timer begins
15:     **end if**
16:     **if** timer ends **then**
17:         **if** intervention is REMIND IN ADVANCE **then**
18:             trigger the intervention
19:         **else if** intervention is NOTIFY IF FORGOTTEN **and** $\hat{s}$ has not been detected by HAR **then**
20:             trigger the intervention
21:         **end if**
22:     **end if**
23: **end while**
24: end the task

---

gauge the total uncertainty the system has, entropy can be calculated as,

$$H[D_t^{\hat{s}}] = -\sum P(D_t^{\hat{s}}) \log P(D_t^{\hat{s}})$$

Higher entropy means more uncertainty, meaning a variance in the estimation of $D_t^{\hat{s}}$. Conversely, a low entropy value indicates that the user behavior is more predictable. To computationally calculate the expectation $E[D_t^{\hat{s}}]$ and entropy $H[D_t^{\hat{s}}]$ in real-time, we used the Monte Carlo method [33] with the sample size of 10,000. Additionally, $T_s^{\hat{s}}$ is enumerated with depth-first search over G from the current step s to the target step $\hat{s}$.

Example distribution of $D_t^{\hat{s}}$ calculated from one session data (cooking task consisting of 14 steps, which we detail in Section 4.1) is shown in Figure 3. In general, the distribution gets more certain over time till the target step because the uncertainties are mitigated by sensor observation. For instance, the estimation gradually becomes highly certain when the target step is s_4 (Figure 3 Left). On the other hand, when the target step is s_{10} or s_{13} (Figure 3 Center&Right), it has a high variance in the beginning primarily because there are several possible paths to the step (majorly whether cooking an egg or a sausage first). After approximately 200 seconds, the variance diminishes when the system detects the step of pouring an egg on the pan, resolving the uncertainty by considering the sensor observation and transition graph.

3.3.2 Intervention Policy.
Lastly, we describe how the intervention is triggered based on the distribution of $E[D_t^{\hat{s}}]$. The key idea is to preemptively forecast the timing of the target step and prepare the
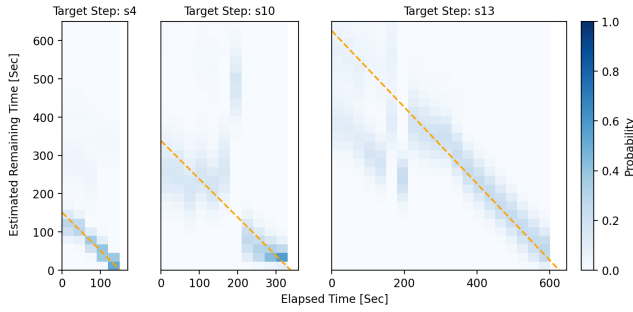
Figure 3: Example $D_t^{\hat{s}}$ transition during a session of the cooking task, where $\hat{s}$ is s_4 (performed early), s_{10} (performed in the middle), and s_{13} (performed later). The orange line is the ground truth (actual remaining time till the target step). Two uncertainties affect the distribution: current belief about the user state and future user trajectory.

intervention. To achieve this, the system continuously characterizes user behavior by $E[D_t^{\hat{s}}]$ and $H[D_t^{\hat{s}}]$. Figure 4 shows the example transitions related to s_7 and s_{13} in different sessions of the cooking task. The expected remaining time (in Figure 4 Top) gets reduced as the user performs each step and gets closer to the target step. The measured entropy can remain quite noisy (as demonstrated in Figure 4 Bottom-Left) due to uncertainties in sensor data and future user behavior. However, we look for brief moments of certainty for the model where it has a reasonable idea of when the target step might happen.

The system starts a timer once the entropy ($H[D_t^{\hat{s}}]$) gets lower than a step-dependent hyperparameter $h^{\hat{s}}$, and $E[D_t^{\hat{s}}]$ is stable for a certain period afterward ("1" in Figure 4). We provide a further detailed implementation of this timer policy in Appendix B. This approach prevents the model from being confused later by sensor uncertainty, especially when the user takes unexpected behavior near the target step. Of course, this policy can still lead to imperfections, but we will quantify its effectiveness for different tasks in Study 1.

Once the timer starts ("2" in Figure 4), the duration of the timer depends on the type of intervention. In case of the REMIND IN ADVANCE intervention, the timer is set for K^- seconds before the anticipated moment for the step of interest. At the end of the timer, the notification is triggered ("3" in Figure 4) to remind the user to make sure they do the step. In case of NOTIFY IF FORGOTTEN intervention, the timer is set for K^+ *after* the anticipated moment for the step of interest. Until the timer runs out, the system waits for the user to do the step of interest. If the user does not perform the step, the system generates a NOTIFY IF FORGOTTEN intervention. The values of constants K^- and K^+ are decided by a system designer or can be adjusted by the end-user. For example, if a user wants a notification as soon as possible after they skip a step, a smaller K^+ is chosen, but it might lead to noisy interactions. We envision designers will experiment with optimal values for these constants to tailor the interventions to their goals.

4 STUDY 1: ALGORITHM EVALUATION IN MULTIPLE DAILY TASKS

We first investigate the effectiveness of the proposed algorithm in preemptively predicting the timing of specific steps. We use a dataset of multiple daily procedures with different task complexity.

4.1 Dataset

We used the dataset of procedural tasks introduced in [8], that is, *latte-making* (22 sessions, 15 participants) and *wound care* (23 sessions, 23 participants) tasks. Following the same protocol, we expanded the dataset by collecting new data on a *cooking* task with 17 sessions and 8 participants who were already familiar with the task. This data collection occurred in a single kitchen. Note that, unlike the latte-making and wound care tasks where the watch was worn on the right wrist, the watch was placed on the left for the cooking task. This difference in the setting led to differences in the motion data patterns since the participants were all right-handed, and thus, the watch might not have captured some crucial motions performed by the dominant hand. Still, we anticipated that this new cooking scenario – featuring users with the watch on their non-dominant hand's wrist – would enhance the system's usability and reflect more natural use cases. We preprocessed and obtained frame-level classification results using PrISM-Tracker [8] with the same frame length of 0.2 seconds, to which we applied our framework.

These three tasks encompass a range of procedural complexity, distinguished by the number of branching paths within each task. Wound care is a training task during perioperative counseling for skin cancer patients in a medical facility. Given the clinical staff trains the patients on how to clean their post-surgical wounds, the task is linear, where the sequence of steps is predetermined. However, given the individual variance in dexterity of patients, there is a high variance in the duration of each step across participants. In contrast, the latte-making task is done by regular users of a coffee machine in an academic building. These users were not given set instructions on how to use the machine. Thus, the task involves more flexibility, allowing for various sequences of steps to be performed with minimal restrictions on ordering. The cooking task represents an intermediate level of complexity. It includes 14 steps where users prepare a sunny-side-up and grilled sausage. The participants can choose the order in which they cook each item, which creates a major branch in the transition. Figure 5 summarizes the transition graphs for the three tasks. Notably, the dataset did not include error cases, and all steps were properly executed. We will test our system's real-time capability to detect errors in Study 2.

4.2 Metric

Study 1 aims to measure the algorithm's accuracy in forecasting the timing of future target steps. To evaluate this, we processed session data from the beginning to emulate real-time prediction. When the intervention timer started at time t according to a policy, we calculated the error between the expected remaining time $E[D_t^{\hat{s}}]$ and the actual remaining time till the target step. The example of this error calculation is presented in Figure 6. Also, while the system will be designed for certain target steps ($\hat{S}$) as discussed
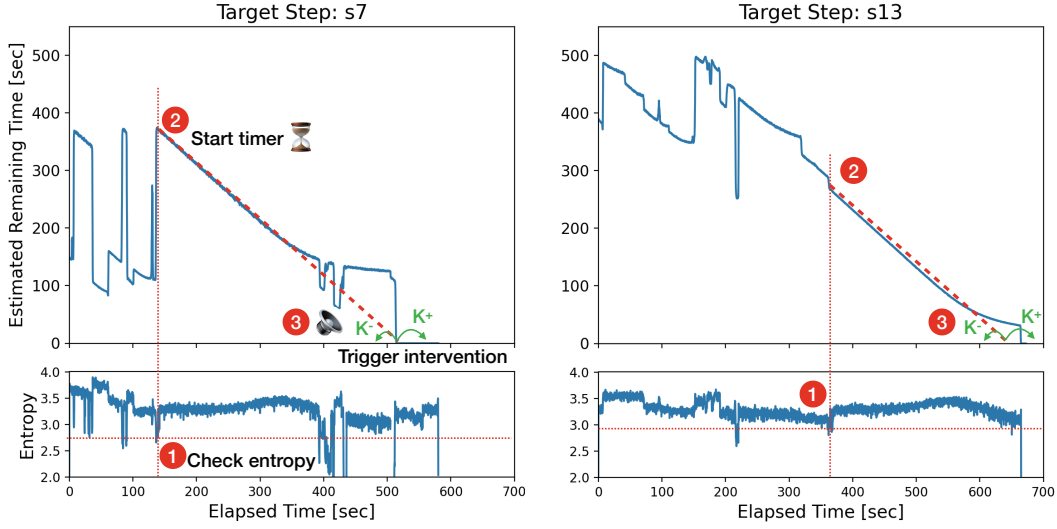
Figure 4: Example transitions of estimated remaining time $E[D_t^{\hat{s}}]$ and entropy $H[D_t^{\hat{s}}]$ from different sessions with different target steps and how our intervention policy works. The y-axis for the entropy graph starts from 2.0 for visualization. K^- and K^+ are system parameters for the timing offsets of the REMIND IN ADVANCE and NOTIFY IF FORGOTTEN interventions, respectively.

in Section 3.2, we assumed $\hat{S} = S$ to examine the accuracy for all steps in this study.

4.3 Compared Intervention Policies

We used the policy described in Section 3.3.2 as the *proposed* condition. We conducted leave-one-session-out cross-validation, and hyperparameters $h^{s_i}(s_i \in S)$ in this policy were obtained by grid search in each training fold.

In addition, we prepared the *baseline* policy, which is based on the expected time to the target step at the beginning, *i.e.*, $E[D_0^{\hat{s}}]$. This policy serves as a condition where the system does not use the sensor data and relies on the transition history to predict the timing of the target step.

4.4 Results

The absolute timing error in the three tasks by different target steps $\hat{s}$ is presented in Figure 7. Overall, the *proposed* policy reduced the timing error from the *baseline* policy at varying target steps, especially in tasks with more complexity. At the same time, we observe the *proposed* policy did not contribute much to reducing the errors at certain steps. We delve deeper into the result of each task and discuss the effect of task complexity and sensing reliability on the error.

4.4.1 Wound Care. Both policies led to small errors (26.8 seconds for the *baseline* policy and 24.1 seconds for the *proposed* policy on average across all steps). The satisfactory performance of the *baseline* policy can be attributed to the single-threaded nature of the task. Furthermore, the tracker's accuracy for this task is limited, as the steps do not introduce significant differences in sensor readings, particularly for intermediate steps ($s_3, ..., s_9$), as illustrated in Figure 12 in Appendix A. The confusion led to the increased

error for s_6 in the *proposed* policy. The result suggests that, when the sensing consists of much uncertainty, the system struggles to model user behavior from the observation. Still, when the sensing accurately detects some actions, it benefits the *proposed* policy. For instance, the tracker's effective detection of state s_2 and s_{11} significantly lowers the error in predicting the subsequent state s_3 and s_{12} in the *proposed* policy.

4.4.2 Cooking. The *proposed* policy largely reduced the timing error compared to the *baseline* policy (119.5 seconds for the *baseline* policy and 61.5 seconds for the *proposed* policy). In the *baseline* policy, the lack of context as to which item to cook first led to a large error, while the *proposed* policy could infer it from the observation. Similarly to the wound care result, it is implied that the larger errors in the *proposed* policy (*i.e.*, s_8, s_9, s_{10}) came from the low sensing accuracy of s_7 and s_8, as shown in Figure 13 in Appendix A, in addition to the relatively longer duration of s_9.

4.4.3 Latte-Making. Lastly, the *proposed* policy again largely reduced the timing error compared to the *baseline* policy (50.1 seconds for the *baseline* policy and 22.3 seconds for the *proposed* policy). The error in the *baseline* policy got larger, especially around the middle steps where different transition branches exist. On the other hand, it can be seen that the *proposed* policy effectively used the sensor data to forecast the timing of different target steps.

4.5 Discussion

The results demonstrated the clear effectiveness of the proposed approach and its implications. First, when tasks are complex and feature multiple potential transition pathways, employing stochastic models to predict user behavior constantly updated by sensor observation proves particularly beneficial. It could be contended that the sequence of steps ought to remain constant, especially for
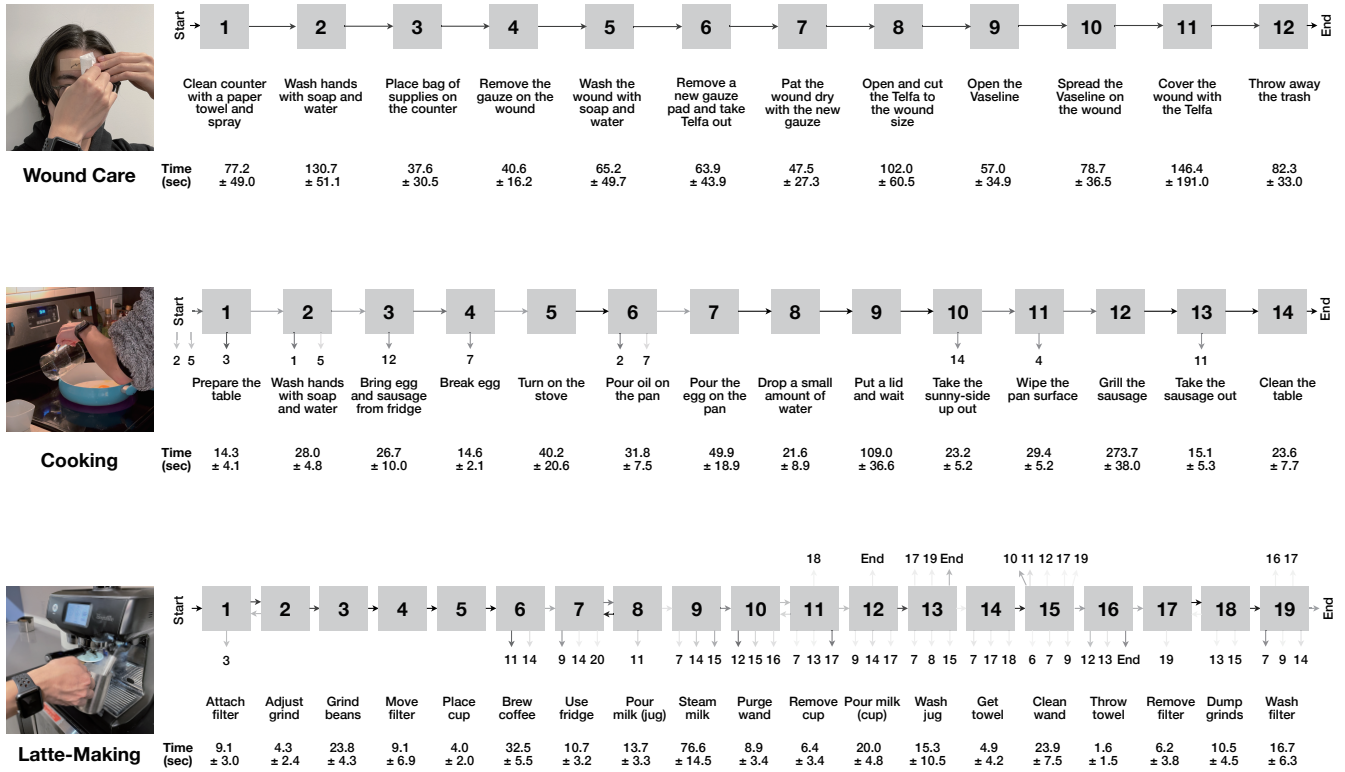
Figure 5: Transition graph of the three tasks: wound care, cooking, and latte-making. Each task has a different level of graph complexity, *i.e.*, the number of branches. Each arrow points to a possible transition step, and the opacity of the arrow represents the probability of the transition.
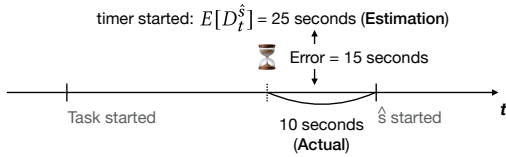

Figure 6: Example error calculation in the forecasted and actual timing of the target step in Study 1.

routine tasks. However, we chose to allow users the flexibility to adjust the order of steps as per their specific circumstances. Such flexibility enables broader user scenarios as discussed in Section 3.1. Simultaneously, enabling the system to learn from the same user's data over days represents a promising direction for future deployment.

The results also corroborated that uncertainty remains. Tracking all steps in complex, everyday tasks with a common device like a smartwatch is quite challenging. The limited tracking accuracy of certain target steps affects the behavior modeling around them. Still, an effective intervention system does not always need to be built on highly accurate sensing. For example, step counters can be inaccurate sensors [49] but still an effective tool for behavior change [40]. Thus, we examine the utility of our intervention design in Study 2.

5 REAL-TIME AGENT SYSTEM IMPLEMENTATION

Study 1 showed the proposed approach's effectiveness in forecasting the timing of target steps, based on which our interventions are triggered. To examine the utility of such real-time interventions, we developed a prototype using Apple Watch (Series 7, watchOS 10.3) with Swift. Currently, there are two versions of the implementation: the laptop-server version and the watch-only version. For the laptop-server version, we used a MacBook Pro with 16GB Apple M1 Chip as a server. The Apple Watch streams the sensor data obtained through dedicated APIs (*i.e.*, AVAudioEngine and CoreMotion) via the network to the server in real-time. The server runs PrISM-Tracker (frame-level HAR and the Viterbi correction), followed by PrISM-Observer (stochastic modeling and intervention policy). When the intervention is to happen, the laptop plays the audio file. This implementation is fast enough because it can utilize efficient computation libraries in Python. We used this version for the user study (Study 2).

At the same time, we developed a watch-only version to show the feasibility of a self-contained agent on the device. The process pipeline is shown in Figure 8. Here, the obtained multimodal data is funneled into a pre-trained feature extractor that operates on the principle of CoreML [4], Apple's machine learning framework. The feature extractor analyzes the incoming data streams and translates
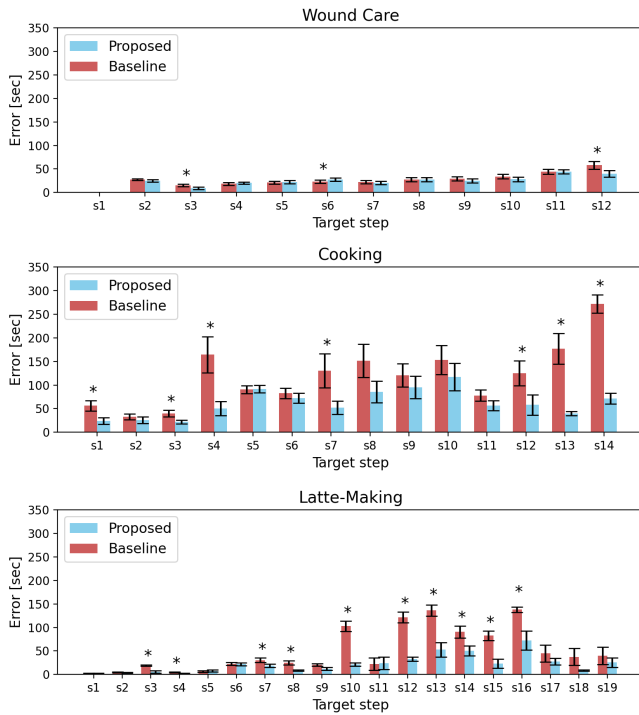
Figure 7: Result of Study 1 comparing the *baseline* and *proposed* policies in three daily procedural tasks. The *proposed* policy greatly reduces the timing error as the procedure's complexity increases thanks to the sensor observation and the introduced stochastic modeling of user behavior. Error bars indicate standard error. * indicates $p < 0.05$ in a two-sided paired t-test.
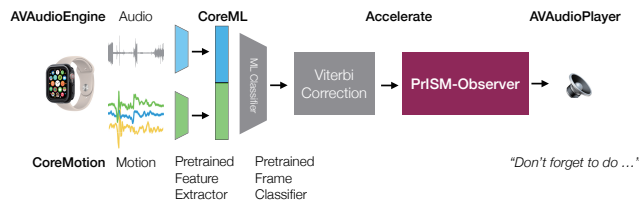

Figure 8: Implementation overview of the real-time agent system on Apple Watch.

them into a feature-rich format suitable for the subsequent frame-level classifier, which CoreML also powers. Note that these models were already trained offline in the same manner with the prior work [8] and converted into CoreML format beforehand using `coremltools`[1]. Following the classifier, the system applies the Viterbi tracking and intervention policy (including the stochastic modeling). These components leverage Apple's Accelerate [3] for high-performance computations. Finally, when the system decides to trigger interventions, the watch plays the corresponding audio file. In both versions, note that while the system emits sound for

[1]https://github.com/apple/coremltools

Table 1: Average time taken to process one frame (equivalent to 0.2 seconds) at each module in our real-time agent system

Module	Laptop (sec)	Watch (sec)
Frame-level HAR	0.04	0.28
Viterbi tracking	0.0002	0.001
Intervention policy	0.09	0.98

intervention, the sensor readings of both motion and audio stop to prevent noise from adding to the tracking.

We conducted a speed test by applying each module to a frame 500 times, the result of which is shown in Table 1. This result shows that the laptop-server version is fast enough to run the pipeline at 5 fps (Recall one frame is 0.2 seconds long). However, the watch-only version is slower due to the current limitation in the computation power and lack of an efficient Swift library. In detail, the most time-consuming computation is the Monte Carlo method to estimate the distribution of $D_t^{\hat{S}}$. Thus, the current watch system runs PrISM-Tracker at 2 fps to update its internal state and PrISM-Observer once per 3 seconds to control intervention, which is still effective, provided each step has a certain duration. We believe future engineering can further accelerate the watch-only version.

6 STUDY 2: USER STUDY WITH A REAL-TIME AGENT SYSTEM IN COOKING

Finally, we conducted a user study to examine the accuracy of the real-time system and users' perception of the performance of in-situ interventions. We used the laptop-server version implementation with the *cooking* task introduced in Study 1, which showed that the task has relatively larger errors in forecasting the target step moment even though the proposed approach mitigates them. This study involves evaluating the intervention effect where such uncertainty remains.

6.1 System Configuration

We first trained PrISM-Tracker [8] (*i.e.*, frame classifier and transition graph G) as the tracking module and obtained the best thresholds ($h^{s_1}, h^{s_2}, ..., h^{s_{14}}$) for the intervention policy using all 17-session training data in the dataset. Moreover, we decided the remaining parameters in the system, that is, K^- and K^+, governing the timing of the REMIND IN ADVANCE and NOTIFY IF FORGOTTEN interventions around the target step, respectively. The authors explored different values before the study and set $K^- = 15$ and $K^+ = 15$ (seconds) as reasonable timing to trigger each type of intervention before and after the moment the target step is supposed to happen.

In addition, we chose candidate steps $\hat{S}$ from all the 14 steps that users may forget or want to be reminded of (Recall the step selection process shown in Figure 2). Here, two system designers individually chose candidate steps first and then had a discussion to reach an agreement. As a result, they chose five steps: s_2 washing hands with soap and water, s_6 pouring oil on the pan, s_8 dropping a small amount of water, s_{11} wiping the pan surface, and s_{14} cleaning the table. Moreover, as described in Section 3.2, the system suggests the intervention type option based on each step's detectability by frame-by-frame HAR (without the Viterbi correction [8], corresponding

Table 2: Candidate intervention steps $\hat{S}$ in Study 2. Their intervention type was decided based on how accurately the step is detected by HAR and input by an external system designer. In actual cases, the selection is customizable based on user preferences.

Step	Error in Study 1 (sec)	HAR Accuracy (F1)	Intervention Type
s_2: washing hands with soap and water	25.9	0.83	NOTIFY IF FORGOTTEN
s_6: pouring oil on the pan	72.3	0.64	REMIND IN ADVANCE
s_8: dropping a small amount of water	85.4	0.28	REMIND IN ADVANCE
s_{11}: wiping the pan surface	56.2	0.43	NOTIFY IF FORGOTTEN
s_{14}: cleaning the table	71.0	0.52	NOTIFY IF FORGOTTEN

Table 3: Participant information and their chosen steps for the intervention in Study 2.

Participant	Gender	Age	Frequency	Environment	Selected Steps	Skipped Steps
P1	M	20's	Occasional	kitchen 1	s_6, s_8, s_{11}	s_{14}
P2	M	20's	Often	kitchen 1	$s_2, s_6, s_8, s_{11}, s_{14}$	
P3	M	20's	Often	kitchen 1	s_8, s_{11}, s_{14}	s_{11}
P4	F	20's	Often	kitchen 1	$s_2, s_6, s_8, s_{11}, s_{14}$	s_2, s_{14}
P5	F	20's	Very often	kitchen 1	s_2, s_{11}, s_{14}	s_{11}, s_{14}
P6	F	40's	Often	kitchen 2	s_2, s_6, s_8, s_{11}	s_2, s_{11}
P7	M	70's	Very often	kitchen 2	s_2, s_8, s_{14}	s_2, s_{14}
P8	F	60's	Very often	kitchen 2	$s_2, s_6, s_8, s_{11}, s_{14}$	s_{11}
P9	M	30's	Occasional	kitchen 2	$s_2, s_6, s_8, s_{11}, s_{14}$	s_2, s_{11}, s_{14}
P10	M	20's	Occasional	kitchen 2	s_8, s_{11}, s_{14}	s_{14}

to the left figure in Figure 13 in Appendix A). The F1-score for the chosen steps were 0.83, 0.64, 0.28, 0.43, and 0.52, respectively. We first decided to assign REMIND IN ADVANCE to s_{11} due to its low accuracy. Moreover, the system designers suggested that doing s_6 before putting an item on the pan is important in actual cooking, and thus, we assigned REMIND IN ADVANCE to s_6 as well. We assigned NOTIFY IF FORGOTTEN to the other steps. Note that we used the same intervention set for all participants in this study to make the comparison meaningful, but the end-users would always have the flexibility to switch the type. The detection accuracy and the assigned intervention type for each $\hat{s} \in \hat{S}$ are summarized in Table 2.

6.2 Procedure

We recruited 10 participants (P1 – P10) through word-of-mouth. They self-reported how often they cook in their daily lives and were all right-handed. To test the robustness against different environments, we used the same kitchen (kitchen 1) as in the training data for five participants and a different kitchen (kitchen 2) for the rest of the participants. Their demographic information is presented in Table 3.

6.2.1 Step Selection. After consent, we showed a list of all the steps S and a tutorial video. Then, we shared interaction candidate steps $\hat{S}$, and asked which steps the users believe they might miss at times and would prefer some intervention support. We asked them to choose at least two steps for the experiment. We also encouraged them to imagine the task would be their routine and which intervention they might want. Simultaneously, we emphasized that they could change the step order flexibly without having to follow the tutorial. We configured the agent system once they finalized the steps.

Here, if their chosen steps included a step for the intervention of NOTIFY IF FORGOTTEN, we asked them to intentionally skip some of them to test the accuracy of the intervention policy. Note, if the participant intentionally skipped the last step s_{14}, the study ran for an extra $K^+ + 23.6$ (the average time for s_{14}) seconds to give the system enough time to judge whether to trigger the NOTIFY IF FORGOTTEN intervention for s_{14}. We did not specify what to do during the extra time, and the participants behaved naturally. The selected steps for each participant and which step they skipped are also shown in Table 3.

6.2.2 Task Execution. Once we configured the system, we answered their questions about the task to ensure they understood the procedure. We also explained that, during the task execution, they were expected not to pause for questions (except in case of an emergency). They pressed the button on the watch app to begin the task and hit the button again to end the task. After the task, they completed a questionnaire, followed by semi-structured interviews about their experience of the system. The entire session took approximately 30 minutes for one participant.

6.3 Metric

We measured the timing difference between the triggered intervention and the target step. Here, the timing for the target step was the moment the participant finished the previous step, which is the same timing as they started the target step unless they intentionally skipped the target step. Note that the triggered intervention timing here accounted for the offset, that is, K^+ and K^-. We manually annotated these timings for each session.

In addition, for the NOTIFY IF FORGOTTEN intervention, we annotated if each intervention was accurate. True positive (TP) when
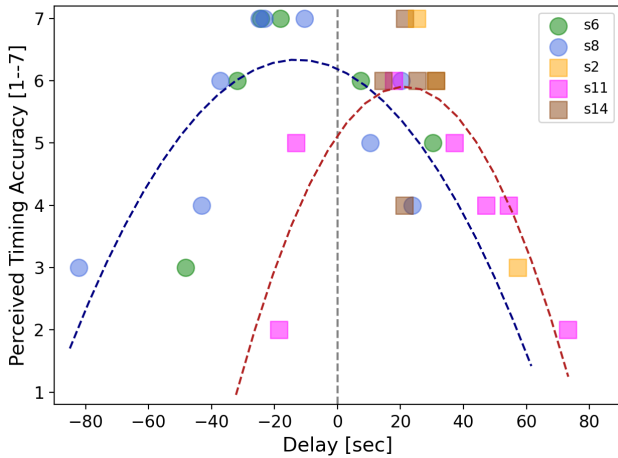
Figure 9: The delay of each triggered intervention from the actual timing of the target step and its perceived timing accuracy. The shape indicates the intervention type (circle: REMIND IN ADVANCE and square: NOTIFY IF FORGOTTEN). The color indicates different steps. The two dashed lines are trend lines obtained by fitting a quadratic function separately to each intervention type (blue: REMIND IN ADVANCE and red: NOTIFY IF FORGOTTEN).
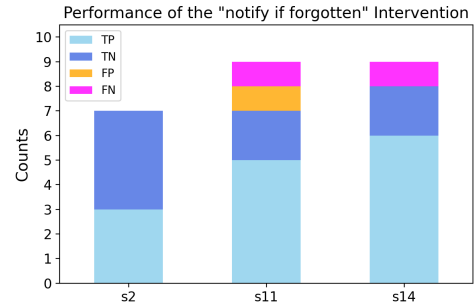

Figure 10: Performance of each of the NOTIFY IF FORGOTTEN interventions. TP, TN, FP, and FN indicate true positive, true negative, false positive, and false negative, respectively.

it was triggered and the participant skipped the target step; false positive (FP) when it was triggered and the participant did the step; false negative (FN) when it was not triggered and the participant skipped the step; and, true negative (TN) when it was not triggered and the participant did the step.

In the post-task questionnaire, we asked the participants to rate the timing accuracy of each intervention they experienced. Note, for the NOTIFY IF FORGOTTEN intervention, we asked about the timing accuracy only if it was triggered (*i.e.*, either TP or FP). They answered whether each intervention was accurate in terms of timing with a Likert scale from 1 (strongly disagree) to 7 (strongly agree). Then, they answered whether the system is reliable with a Likert scale from 1 (strongly disagree) to 7 (strongly agree). In addition, we asked for their behavioral intention, guided by the Technology Acceptance Model [15], which explains users' attitudes towards technologies and is frequently used to evaluate how likely individuals are to use the technologies.

6.4 Results

Figure 9 shows the delay of each intervention and how accurate the participants felt about the timing of the intervention. The circle markers indicate the REMIND IN ADVANCE intervention (*i.e.*, s_6 and s_8) and the square markers indicate the NOTIFY IF FORGOTTEN intervention (*i.e.*, s_2, s_{11}, and s_{14}). The result suggests that the agent system could trigger most interventions with a small timing error, and the participants felt they were accurate (20 out of the total 27 triggered interventions were rated five or higher on the 1–7 Likert scale). Also, by looking at the trend lines, the relationship between

the delay and perceived accuracy is implied. For instance, the participants perceived the REMIND IN ADVANCE intervention (circle markers) as inaccurate as the delay becomes positively larger. At the same time, it was best rated when triggered some seconds before the delay is 0. This is aligned with the type of the intervention—users want it to be triggered before the target step—validating our design, specifically, the use of the offset K^- with $E[D_t^{\hat{s}}]$. Conversely, they rated higher for the NOTIFY IF FORGOTTEN intervention (square markers) if they happen some seconds after, also corroborating the use of the offset K^+. While more samples are demanded to examine the trend, we can infer that the participants' expectations about the different interventions affected their perception of the intervention timing.

Additionally, there was no significant difference in the delay between kitchens 1 and 2 ($p > 0.05$) in this cooking scenario. While more investigation is demanded to quantify the effect, this result suggests the benefit of the smartwatch-based agent system: the location-independent HAR capability.

Figure 10 shows whether each of the NOTIFY IF FORGOTTEN interventions was accurate. While the number of samples is small, the results of the low error rate (3 out of 25 chances) indicate that the agent system could detect whether the target step happened or not and notify the participants appropriately. More specifically, there was no false intervention for s_2 thanks to the high HAR accuracy (See Table 2). On the other hand, there were false interventions for s_{11} and s_{14}, which can be explained by a relatively lower HAR accuracy. Given this, the agent system could infer the risk of false interventions in the model training phase, which would help system designers configure the system.

Figure 11 presents the relationship between the overall reliability and the behavioral intention answered by the participants. While a longer-term study is ideal, this shows initial evidence that such a task-support agent system is favorably accepted. The result also suggests a correlation between reliability and behavioral intention, which implies that reducing errors and achieving situated interventions is important.

6.5 User Comments

Finally, we summarized the qualitative insights from the semi-structured interviews after the task.
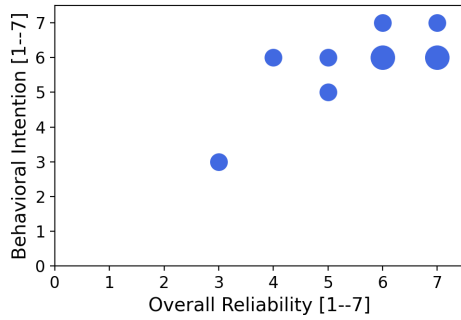
Figure 11: The rated reliability and behavioral intention of the agent system. Each dot represents a participant's answer to the questionnaire. The larger circles indicate two participants.

6.5.1 How Participants Selected Interventions. The number of participants who enabled each target step $\hat{s} \in \hat{S}$ in the step selection phase is 7, 6, 8, 9, and 5, for s_2, s_6, s_8, s_{11}, and s_{14}, respectively. The individual difference shown in Table 3 reflects their preference. For instance, P7 mentioned, "I never forget pouring oil when using the pan, so not including s_2." P3 said, "I thought I would be inattentive in the latter part of the procedure." P5 said, "I'm experienced. I only need them if I make a mistake." P6 commented, "I am a careless person. I would enable all notifications. I believe I would not be annoyed very much. It'd be like a fun assistant." In addition to these comments, they agreed that using different types of intervention made sense. P8 said, "I don't need to be reminded every time to wash hands, so the system notifying me only if I make a mistake is a good design." The results underpin the design of our framework, enabling end-user customization of the intervention.

6.5.2 How Participants Perceived Interventions. Next, we asked about the interventions they remembered most and their impression of the interventions. They were surprised by the agent's capability, especially for the NOTIFY IF FORGOTTEN intervention. P1 said, "I'm amazed by how it knows about my situation. The notification (for s_{14}) was accurate." At the same time, a few participants mentioned a need for certain notifications to be more precise. P9 mentioned, "For s_{11} (wiping the pan surface), it was after I already put my sausage, and thus I felt it is a bit inaccurate." P3 also suggested that it is safe to trigger interventions earlier as the steps can be irreversible in the cooking. Our framework is flexible enough to incorporate such adjustments, for example, by using a smaller K^+ value to enable immediate detection or changing the type of the intervention into REMIND IN ADVANCE.

Similarly, regarding the NOTIFY IF FORGOTTEN intervention, P2, after experiencing FP (false positive), said, "I was not annoyed much by the notification (s_{11}). I just thought it was checking me." On the other hand, P1 experienced FN (false negative), mentioning, "I was a bit sad when it did not happen even though I skipped it (s_{14}). In the real world, strict checking may be good for me." The results suggest the tendency of false-positive resistance and false-negative sensitiveness, consistent with a design strategy recognized in research on real-time interventions that rely on imperfect sensing [7].

Moreover, the participants agreed that the REMIND IN ADVANCE intervention is useful for unfamiliar steps like s_8. They also suggested that it should happen before the step. P9 mentioned, "Earlier is good. If it is after I have done the step, it would sound hilarious. But too early is also not good. Being able to feel some intelligence in the system is what attracts me." This comment highlights the importance of minimizing timing errors and the design of interventions that meet user expectations.

6.5.3 Further Use Cases. Acknowledging the capability of the agent system, the participants shared an interest in extending the application. The examples include skin care (3 participants), gym/exercise routine (3 participants), furniture assembly (2 participants), medication (2 participants), house cleaning (2 participants), using a laundry room (1 participant), and using a maker space (1 participant). For example, P1 mentioned, "I sometimes forget to sanitize the machine after I use it in the gym. Also, it would help me complete my exercise routine without forgetting an activity." Supporting these various needs is promising, given PrISM-Observer's generalizable performance suggested in Study 1.

6.5.4 Room for Improvement. Finally, we also gained insights about future improvement. P8 said, "The system did not work well for me throughout the task. It was wrong from the beginning. I wanted to fix it by telling it." For this user, the system mistook the path P8 followed; P8 cooked the sausage first while the system guessed she cooked an egg first due to HAR error, resulting in overly early interventions. In such a case, as P8 expressed, offering a way for the user to correct the agent's belief will be ideal, which we discuss further later in Section 7.5.

On another note, P1 mentioned, "I would like to see more variations in the audio message with different voice styles. If it were my favorite idol's voice, I might accept it even if the intervention is wrong." Such a social aspect between the task-support agent system and the user is an exciting research area [45].

6.6 Summary

The results demonstrated the accuracy of our agent system's intervention, which the participants favorably accepted. The user comments highlighted the benefits of PrISM-Observer's design, such as the flexibility to cater to different user needs and to adjust parameters to enhance the user experience. We believe Study 2 produces insights into the human-AI interaction system powered by sensing technology in the physical world.

It should be noted that although seven participants completed the task in a sequence differing from that presented in the tutorial, we cannot rule out the possibility of a study-induced bias affecting task behavior. Acknowledging this limitation, we plan to undertake a longer-term study to assess the system's ability to adapt to spontaneous user behavior during tasks.

7 LIMITATIONS AND FUTURE WORK

The current formulation and implementation of PrISM-Observer is not without limitations. We also describe future work to advance the field further.

7.1 Long-Term Behavior Study

Study 2 is in a controlled setting and thus does not perfectly reflect the user behavior performing routine tasks under natural conditions, such as inattentiveness. A long-term real-world study is needed to further investigate the system's benefits and user perception. We want to emphasize that user behavior should be considered flexible instead of a fixed sequence for reasonable human-AI interaction so the system can adjust to spontaneous user behavior, such as switching the order of the steps. We believe PrISM-Observer's capability to incorporate such uncertainty in human behavior will be a fundamental solution.

In addition, a long-term study will also shed light on the agent's capability to learn from user behavior to adapt the intervention policy. For instance, if the user often forgets a certain step, the system switches the intervention from NOTIFY-IF-FORGOTTEN to REMIND-IN-ADVANCE. Conversely, if the user gets accustomed to the task enough and thinks REMIND-IN-ADVANCE is too much, it can be switched to NOTIFY-IF-FORGOTTEN. It is of interest to examine the optimal balance of the agent's adaptability and the end-user's controllability.

7.2 Health and Accessibility Applications

The proposed framework will be especially helpful for health applications to assist people in need. For example, people with dementia struggle to perform everyday routine tasks properly, and thus, interventions are helpful [24, 28]. Moreover, patients who need to perform self-care after surgery regularly could benefit from it to avoid the critical consequence of infection due to mistakes [30, 48]. We are currently developing and evaluating assistants for postoperative wound care with skin-cancer patients undergoing Mohs micrographic surgery [51].

7.3 Detecting Errors within a Step

PrISM-Observer can detect and remind a user of a step in the procedure. Currently, it does not support dealing with more fine-grained errors, such as errors within a step, often necessitating visual information to be detected. For example, while steaming milk for the latte-making task, it is important to keep the jug at a proper angle to get a good milk texture, which the current system cannot sense. While aligning user expectations about the system's capability is crucial, we plan to investigate further sensing capability, such as applying multimodal anomaly detection [6] to compare the sensor data within the step with prior "good" behavior.

7.4 Refining Step Granularity

Automating and optimizing the process of dividing the procedure into steps is crucial to scaling our framework to diverse tasks. In this regard, Zhou *et al.* [57] proposed an approach to constructing an open-domain hierarchical knowledge base of procedures. Wake *et al.* [52] created a task model for decomposing human demonstration of procedural tasks for robots to model the process. In our case, it could be possible for the framework to use an automatically generated transition graph internally to model the user behavior and to trigger intervention. In contrast, the user specifies the desired intervention without being aware of the representation of the steps used by the system.

7.5 Interweaving Broader Interactions

This work focuses on passive interaction from the user's perspective; the system monitors user task behavior and proactively triggers intervention. Though our study demonstrated its effectiveness, the potential role of a task-support intelligent agent can be more versatile [23]. For instance, more dialogue between the user and the system could be an interesting area to study. PrISM-Tracker [8] showed a preliminary approach to update its tracking belief through dialogue. In the framework of PrISM-Observer, if the system could ask the user "What are you doing?", the system could resolve the uncertainty about the current step. Similarly, asking "What will you do next?" would resolve the uncertainty about the future step transition. Conversely, the user's reaction to the intervention could correct errors in the agent, as suggested in Section 6.5.4. Likewise, if there is a question-answering capability, the user's asking, "What should I do after washing my hands?" would also help the agent track the user's state. Given the surge of human-like chat capability enabled by large language models, investigating such real-time dialogue interactions along with sensor data is promising.

7.6 Extending to Other Sensing Platforms

We implemented our prototype on a smartwatch based on its capability to sense a user across various daily activities at different places. At the same time, the framework's stochastic modeling can be used as a post-process for other HAR systems. For example, *VAX* [37] uses ambient, privacy-sensitive sensors such as Doppler RADARs and LIDARs. We can install such a HAR system in the user's kitchen to monitor their everyday cooking and use PrISM-Observer on top of it. Existing voice assistants like Alexa could also be a platform for integrating our framework, leveraging acoustic sensing to obtain user context. In the context of voice assistant, while prior research [14, 16] has shown the benefit of dialogue-based guidance of steps, our work extends their capability by context awareness and user-centered intervention design, as recently emphasized by Jaber *et al.* [23]. Investigating different input sensors and their effect on end-user experience is an important future study.

8 CONCLUSION

We presented *PrISM-Observer*, a framework for designing interventions to mitigate errors in daily procedural tasks (*e.g.*, forgetting a step), and developed a real-time agent system on a smartwatch leveraging multimodal Human Activity Recognition (HAR). The stochastic modeling of user behavior and intervention policy based on it enables situated triggering of interventions. Moreover, the framework is designed so users or system designers can customize which step they need intervention for and how. Study 1, involving three daily task datasets, verified the proposed approach's effectiveness in optimizing the intervention timing. In addition, Study 2, using the real-time smartwatch system in the cooking scenario, resulted in positive participant feedback, providing qualitative insights into such real-time intervention in procedural tasks. The results and discussion pave the way for achieving reliable human-agent interaction where the agent's sensing capability is limited, which is often the case in real-world tasks.

REFERENCES

[1] 2013. Memory Lapse – Four Things Slip Our Mind Every Day. https://en.paperblog.com/memory-lapse-four-things-slip-our-mind-every-day-639158/

[2] Saleema Amershi, Daniel S. Weld, Mihaela Vorvoreanu, Adam Fourney, Besmira Nushi, Penny Collisson, Jina Suh, Shamsi T. Iqbal, Paul N. Bennett, Kori Inkpen, Jaime Teevan, Ruth Kikin-Gil, and Eric Horvitz. 2019. Guidelines for human-AI interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 3. https://doi.org/10.1145/3290605.3300233

[3] Apple. 2022. Accelerate. https://developer.apple.com/documentation/accelerate

[4] Apple. 2022. CoreML. https://developer.apple.com/documentation/coreml

[5] Apple. 2022. Handwashing on Apple Watch. https://support.apple.com/guide/watch/set-up-handwashing-apdc9b9f04a8/watchos

[6] Riku Arakawa and Hiromu Yakura. 2019. REsCUE: A framework for REal-time feedback on behavioral CUEs using multimodal anomaly detection. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems, CHI 2019, Glasgow, Scotland, UK, May 04-09, 2019*. ACM, 572. https://doi.org/10.1145/3290605.3300802

[7] Riku Arakawa and Hiromu Yakura. 2021. Mindless Attractor: A False-Positive Resistant Intervention for Drawing Attention Using Auditory Perturbation. In *CHI '21: CHI Conference on Human Factors in Computing Systems, Virtual Event / Yokohama, Japan, May 8-13, 2021*. ACM, 99:1–99:15. https://doi.org/10.1145/3411764.3445339

[8] Riku Arakawa, Hiromu Yakura, Vimal Mollyn, Suzanne Nie, Emma Russell, Dustin P. DeMeo, Haarika A. Reddy, Alexander K. Maytin, Bryan T. Carroll, Jill Fain Lehman, and Mayank Goel. 2022. PrISM-Tracker: A Framework for Multimodal Procedure Tracking Using Wearable Sensors and State Transition Information with User-Driven Handling of Errors and Uncertainty. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 4 (2022), 156:1–156:27. https://doi.org/10.1145/3569504

[9] Sara Ashry, Tetsuji Ogawa, and Walid Gomaa. 2020. CHARM-Deep: Continuous Human Activity Recognition Model Based on Deep Neural Network Using IMU Sensors of Smartwatch. *IEEE Sensors Journal* 20, 15 (Aug. 2020), 8757–8770. https://doi.org/10.1109/jsen.2020.2985374

[10] Jenna Beaver, Kaci B. Wilson, and Maureen Schmitter-Edgecombe. 2017. Characterising omission errors in everyday task completion and cognitive correlates in individuals with mild cognitive impairment and dementia. *Neuropsychological Rehabilitation* 29, 5 (June 2017), 804–820. https://doi.org/10.1080/09602011.2017.1337039

[11] Vincent Becker, Linus Fessler, and Gábor Sörös. 2019. GestEar: combining audio and motion sensing for gesture recognition on smartwatches. In *Proceedings of the 23rd International Symposium on Wearable Computers, UbiComp 2019, London, UK, September 09-13, 2019*. ACM, 10–19. https://doi.org/10.1145/3341163.3347735

[12] Riccardo Bovo, Nicola Binetti, Duncan P. Brumby, and Simon Julier. 2020. Detecting errors in pick and place procedures: detecting errors in multi-stage and sequence-constrained manual retrieve-assembly procedures. In *IUI '20: 25th International Conference on Intelligent User Interfaces, Cagliari, Italy, March 17-20, 2020*. ACM, 536–545. https://doi.org/10.1145/3377325.3377497

[13] Michael D. Byrne and Susan Bovair. 1997. A Working Memory Model of a Common Procedural Error. *Cognitive Science* 21, 1 (Jan. 1997), 31–61. https://doi.org/10.1207/s15516709cog2101_2

[14] Jiawen Chu. 2021. Recipe Bot: The Application of Conversational AI in Home Cooking Assistant. In *2021 2nd International Conference on Big Data; Artificial Intelligence; Software Engineering (ICBASE)*. IEEE. https://doi.org/10.1109/icbase53849.2021.00136

[15] Fred D. Davis. 1989. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly* 13, 3 (1989), 319–340. https://doi.org/10.2307/249008

[16] Alexander Frummet, Alessandro Speggiorin, David Elsweiler, Anton Leuski, and Jeff Dalton. 2024. Cooking with Conversation: Enhancing User Engagement and Learning with a Knowledge-Enhancing Assistant. *ACM Transactions on Information Systems* (2024).

[17] Wayne D. Gray and Deborah A. Boehm-Davis. 2000. Milliseconds matter: An introduction to microstrategies and to their use in describing and predicting interactive behavior. *Journal of Experimental Psychology: Applied* 6, 4 (2000), 322–335. https://doi.org/10.1037/1076-898x.6.4.322

[18] Yu Guan and Thomas Plötz. 2017. Ensembles of Deep LSTM Learners for Activity Recognition using Wearables. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 1, 2 (2017), 11:1–11:28. https://doi.org/10.1145/3090076

[19] Reiko Hamada, Jun Okabe, Ichiro Ide, Shin'ichi Satoh, Shuichi Sakai, and Hidehiko Tanaka. 2005. Cooking navi: assistant for daily cooking in kitchen. In *Proceedings of the 13th ACM International Conference on Multimedia, Singapore, November 6-11, 2005*. ACM, 371–374. https://doi.org/10.1145/1101149.1101228

[20] Hiroki Hasada, Junjian Zhang, Kenta Yamamoto, Bektur Ryskeldiev, and Yoichi Ochiai. 2019. AR Cooking: Comparing Display Methods for the Instructions of Cookwares on AR Goggles. In *Human Interface and the Management of Information. Information in Intelligent Systems - Thematic Area, HIMI 2019, Held as Part of the 21st HCI International Conference, HCII 2019, Orlando, FL, USA, July 26-31, 2019, Proceedings, Part II (Lecture Notes in Computer Science, Vol. 11570)*. Springer, 127–140. https://doi.org/10.1007/978-3-030-22649-7_11

[21] Jinhui Hu, Cong Xin, Manman Zhang, and Youzhen Chen. 2023. The effect of cognitive load and time stress on prospective memory and its components. *Current Psychology* 43, 2 (Feb. 2023), 1670–1684. https://doi.org/10.1007/s12144-023-04354-1

[22] Gaoping Huang, Xun Qian, Tianyi Wang, Fagun Patel, Maitreya Sreeram, Yuanzhi Cao, Karthik Ramani, and Alexander J. Quinn. 2021. AdapTutAR: An adaptive tutoring system for machine tasks in augmented reality. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 417:1–417:15. https://doi.org/10.1145/3411764.3445283

[23] Razan Jaber, Sabrina Zhong, Sanna Kuoppamäki, Aida Hosseini, Iona Gessinger, Duncan P. Brumby, Benjamin R. Cowan, and Donald McMillan. 2024. Cooking With Agents: Designing Context-aware Voice Interaction. In *Proceedings of the CHI Conference on Human Factors in Computing Systems, CHI 2024, Honolulu, HI, USA, May 11-16, 2024*. ACM, 551:1–551:13. https://doi.org/10.1145/3613904.3642183

[24] Staffan Josephsson, Lars Bäckman, Lena Borell, Birgitta Bernspång, Louise Nygård, and Lisa Rönnberg. 1993. Supporting everyday activities in dementia: An intervention study. *International Journal of Geriatric Psychiatry* 8, 5 (May 1993), 395–400. https://doi.org/10.1002/gps.930080505

[25] Giulio Lancioni, Lorenzo Desideri, Nirbhay Singh, Mark O'Reilly, and Jeff Sigafoos. 2021. Technology options to help people with dementia or acquired cognitive impairment perform multistep daily tasks: a scoping review. *Journal of Enabling Technologies* 15, 3 (May 2021), 208–223. https://doi.org/10.1108/jet-11-2020-0048

[26] Gierad Laput, Karan Ahuja, Mayank Goel, and Chris Harrison. 2018. Ubicoustics: Plug-and-play acoustic activity recognition. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. ACM, New York, NY, 213–224. https://doi.org/10.1145/3242587.3242609

[27] Gierad Laput, Robert Xiao, and Chris Harrison. 2016. ViBand: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*. ACM, New York, NY, 321–333. https://doi.org/10.1145/2984511.2984582

[28] Ioulietta Lazarou, Anastasios Karakostas, Thanos G. Stavropoulos, Theodoros Tsompanidis, Georgios Meditskos, Ioannis Kompatsiaris, and Magda Tsolaki. 2016. A Novel and Intelligent Home Monitoring System for Care Support of Elders with Cognitive Impairment. *Journal of Alzheimer's Disease* 54, 4 (Oct. 2016), 1561–1591. https://doi.org/10.3233/jad-160348

[29] Matthew L. Lee and Anind K. Dey. 2014. Real-time feedback for improving medication taking. In *CHI Conference on Human Factors in Computing Systems, CHI'14, Toronto, ON, Canada - April 26 - May 01, 2014*. ACM, 2259–2268. https://doi.org/10.1145/2556288.2557210

[30] Lowell S Levin and Ellen L Idler. 1983. Self-care in health. *Annual review of public health* 4, 1 (1983), 181–201.

[31] Daniyal Liaqat, Robert Wu, Andrea Gershon, Hisham Alshaer, Frank Rudzicz, and Eyal de Lara. 2018. Challenges with real-world smartwatch based audio monitoring. In *Proceedings of the 4th ACM Workshop on Wearable Systems and Applications, WearSys@MobiSys 2018, Munich, Germany, June 10, 2018*. ACM, 54–59. https://doi.org/10.1145/3211960.3211977

[32] Ziyi Liu, Zhengzhe Zhu, Enze Jiang, Feichi Huang, Ana M. Villanueva, Xun Qian, Tianyi Wang, and Karthik Ramani. 2023. InstruMentAR: Auto-Generation of Augmented Reality Tutorials for Operating Digital Instruments Through Recording Embodied Demonstration. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, CHI 2023, Hamburg, Germany, April 23-28, 2023*. ACM, 32:1–32:17. https://doi.org/10.1145/3544548.3581442

[33] Nicholas Metropolis and Stanislaw Ulam. 1949. The monte carlo method. *Journal of the American statistical association* 44, 247 (1949), 335–341.

[34] Vimal Mollyn, Karan Ahuja, Dhruv Verma, Chris Harrison, and Mayank Goel. 2022. SAMoSA: Sensing Activities with Motion and Subsampled Audio. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 6, 3 (2022), 132:1–132:19. https://doi.org/10.1145/3550284

[35] Yasushi Nakauchi, Takuo Suzuki, Akira Tokumasu, and Sho Murakami. 2009. Cooking procedure recognition and inference in sensor embedded kitchen. In *Proceedings of the 18th IEEE International Symposium on Robot and Human Interactive Communication*. IEEE, New York, NY, 593–600. https://doi.org/10.1109/ROMAN.2009.5326050

[36] Denise C. Park, Roger W. Morrell, David Frieske, and Deborah Kincaid. 1992. Medication adherence behaviors in older adults: Effects of external cognitive supports. *Psychology and Aging* 7, 2 (1992), 252–256. https://doi.org/10.1037/0882-7974.7.2.252

[37] Prasoon Patidar, Mayank Goel, and Yuvraj Agarwal. 2023. VAX: Using Existing Video and Audio-based Activity Recognition Models to Bootstrap Privacy-Sensitive Sensors. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 3 (2023), 117:1–117:24. https://doi.org/10.1145/3610907

[38] Rohith Peddi, Shivvrat Arya, Bharath Challa, Likhitha Pallapothula, Akshay Vyas, Jikai Wang, Qifan Zhang, Vasundhara Komaragiri, Eric D. Ragan, Nicholas Ruozzi, Yu Xiang, and Vibhav Gogate. 2023. CaptainCook4D: A dataset for

understanding errors in procedural activities. *CoRR* abs/2312.14556 (2023). https://doi.org/10.48550/ARXIV.2312.14556 arXiv:2312.14556

[39] Meghana Ratna Pydi, Petra Stankard, Neha Parikh, Purnima Ranawat, Ravneet Kaur, AG Shankar, Angela Chaudhuri, Sonjelle Shilton, Aditi Srinivasan, Joyita Chowdhury, and Elena Ivanova Reipold. 2023. Assessment of the Usability of SARS-CoV-2 Self Tests in a Peer-Assisted Model among Factory Workers in Bengaluru, India. (Nov. 2023). https://doi.org/10.1101/2023.11.20.23298784

[40] Shanhu Qiu, Xue Cai, Xiang Chen, Bingquan Yang, and Zilin Sun. 2014. Step counter use in type 2 diabetes: a meta-analysis of randomized controlled trials. *BMC medicine* 12 (2014), 1–9.

[41] James Reason. 1990. *Human error*. Cambridge university press.

[42] A. Riedel, J. Gerlach, M. Dietsch, S. Herbst, F. Engelmann, N. Brehm, and T. Pfeifroth. 2021. A deep learning-based worker assistance system for error prevention: Case study in a real-world manual assembly. *Advances in Production Engineering & Management* 16, 4 (Dec. 2021), 393–404. https://doi.org/10.14743/apem2021.4.408

[43] Ayaka Sato, Keita Watanabe, and Jun Rekimoto. 2014. MimiCook: A cooking assistant system with situated guidance. In *Proceedings of the 8th International Conference on Tangible, Embedded, and Embodied Interaction*. ACM, New York, NY, 121–124. https://doi.org/10.1145/2540930.2540952

[44] Tim J Schoonbeek, Tim Houben, Hans Onvlee, Fons van der Sommen, et al. 2024. IndustReal: A Dataset for Procedure Step Recognition Handling Execution Errors in Egocentric Videos in an Industrial-Like Setting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 4365–4374.

[45] Katie Seaborn, Norihisa P. Miyake, Peter Pennefather, and Mihoko Otake-Matsuura. 2022. Voice in Human-Agent Interaction: A Survey. *ACM Comput. Surv.* 54, 4 (2022), 81:1–81:43. https://doi.org/10.1145/3386867

[46] Fadime Sener, Dibyadip Chatterjee, Daniel Shelepov, Kun He, Dipika Singhania, Robert Wang, and Angela Yao. 2022. Assembly101: A Large-Scale Multi-View Video Dataset for Understanding Procedural Activities. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*. IEEE, 21064–21074. https://doi.org/10.1109/CVPR52688.2022.02042

[47] J. Serván, F. Mas, J.L. Menéndez, and J. Ríos. 2012. Assembly Work Instruction Deployment Using Augmented Reality. *Key Engineering Materials* 502 (Feb. 2012), 25–30. https://doi.org/10.4028/www.scientific.net/kem.502.25

[48] Carrie L Shandra and Nihil Sonalkar. 2016. Health self-care in the United States. *Public Health* 138 (2016), 26–32. https://doi.org/10.1016/j.puhe.2016.02.030

[49] Charlotte Brun Thorup, Jan Jesper Andreasen, Erik Elgaard Sørensen, Mette Grønkjær, Birthe Irene Dinesen, and John Hansen. 2017. Accuracy of a step counter during treadmill and daily life walking by healthy adults and patients with cardiac disease. *BMJ open* 7, 3 (2017), e011742.

[50] Daisuke Uriu, Mizuki Namai, Satoru Tokuhisa, Ryo Kashiwagi, Masahiko Inami, and Naohito Okude. 2012. Panavi: Recipe medium with a sensors-embedded pan for domestic users to master professional culinary arts. In *Proceedings of the 2012 CHI Conference on Human Factors in Computing Systems*. ACM, New York, NY, 129–138. https://doi.org/10.1145/2207676.2207695

[51] Annalise Vaccarello, Alexander K. Maytin, Yash Kumar, Toluwalashe Onamusi, Haarika A. Reddy, Mayank Goel, Riku Arakawa, Jill Fain Lehman, and Bryan T. Carroll. 2024. Barriers to use of digital assistance for postoperative wound care: a single-center survey of dermatologic surgery patients. *Archives of Dermatological Research* 316, 7 (June 2024). https://doi.org/10.1007/s00403-024-03025-w

[52] Naoki Wake, Riku Arakawa, Iori Yanokura, Takuya Kiyokawa, Kazuhiro Sasabuchi, Jun Takamatsu, and Katsushi Ikeuchi. 2021. A Learning-from-Observation Framework: One-Shot Robot Teaching for Grasp-Manipulation-Release Household Operations. In *IEEE/SICE International Symposium on System Integration, SII 2021, Iwaki, Japan, January 11-14, 2021*. IEEE, 461–466. https://doi.org/10.1109/IEEECONF49454.2021.9382750

[53] Xin Wang, Taein Kwon, Mahdi Rad, Bowen Pan, Ishani Chakraborty, Sean Andrist, Dan Bohus, Ashley Feniello, Bugra Tekin, Felipe Vieira Frujeri, Neel Joshi, and Marc Pollefeys. 2023. HoloAssist: an Egocentric Human Interaction Dataset for Interactive AI Assistants in the Real World. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*. IEEE, 20213–20224. https://doi.org/10.1109/ICCV51070.2023.01854

[54] Santosh Kumar Yadav, Kamlesh Tiwari, Hari Mohan Pandey, and Ali Akbar Shaikh. 2021. A review of multimodal human activity recognition with special emphasis on classification, applications, challenges and future directions. *Knowl. Based Syst.* 223 (2021), 106970. https://doi.org/10.1016/J.KNOSYS.2021.106970

[55] Masahiro Yamaguchi, Shohei Mori, Peter Mohr, Markus Tatzgern, Ana Stanescu, Hideo Saito, and Denis Kalkofen. 2020. Video-Annotated Augmented Reality Assembly Tutorials. In *UIST '20: The 33rd Annual ACM Symposium on User Interface Software and Technology, Virtual Event, USA, October 20-23, 2020*. ACM, 1010–1022. https://doi.org/10.1145/3379337.3415819

[56] Yiran Zhang. 2023. The underlying reasons for making reminders: An investigation on memory offloading from the perspective of cognitive psychology. *Journal of Education, Humanities and Social Sciences* 8 (Feb. 2023), 2208–2213. https://doi.org/10.54097/ehss.v8i.4678

[57] Shuyan Zhou, Li Zhang, Yue Yang, Qing Lyu, Pengcheng Yin, Chris Callison-Burch, and Graham Neubig. 2022. Show Me More Details: Discovering Hierarchies of Procedures from Semi-structured Web Data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*. Association for Computational Linguistics, 2998–3012. https://doi.org/10.18653/V1/2022.ACL-LONG.214

A PRISM-TRACKER'S PERFORMANCE

Figure 12 – Figure 14 show the frame-level HAR confusion matrix of PrISM-Tracker [8] without and with the Viterbi correction, for the wound care, cooking, and latte-making task, respectively. Recall that one frame corresponds to 0.2 seconds. While the Viterbi correction improved the classification performance, inaccuracy remains, especially for steps with similar signal profiles: macro F1-Scores for the wound care, cooking, and latte-making tasks were 50.7%, 61.5%, and 52.9%, respectively.

B DETAILS OF INTERVENTION POLICY IMPLEMENTATION

The intervention policy described in Section 3.3.2 includes several hyperparameters. First, there is a step-dependent entropy threshold $h^{\hat{s}}$, which governs the timing to initiate an intervention timer. Then, after the intervention timer starts, the framework keeps monitoring $E[D_t^{\hat{s}}]$. If there is a significant change within the next $p = 10$ seconds (before the timer ends), the timer is discarded, for which we use a threshold parameter $e = 30$ seconds. Moreover, as shown in Figure 4 Bottom, the entropy fluctuates due to the randomness in the Monte-Calro method, thereby necessitating the need for smoothing. We apply a moving average smoothing with the size of $w = 2$ seconds to the entropy. Parameters p, e, and w were empirically determined in the initial observation of $D_t^{\hat{s}}$ transitions of a few sessions. On the other hand, the entropy threshold $h^{\hat{s}}$ was optimized through grid search in the process of the leave-one-session-out cross-validation.

In addition, for the NOTIFY IF FORGOTTEN intervention in the real-time system used for Study 2 (cooking), the system monitors the user action from the moment the timer started for $E[D_t^{\hat{s}}] + K^+$ seconds to judge whether the target step $\hat{s}$ happens. Since frame-level prediction can fluctuate due to sensing noise, we apply a moving average smoothing with the size of 1 second, which corresponds to 5 frames. Moreover, the system judges the step $\hat{s}$ happens if the probability for the step is highest for 5 seconds, which was also determined empirically based on the fact that every step in the cooking usually lasts more than 10 seconds (See Figure 5).
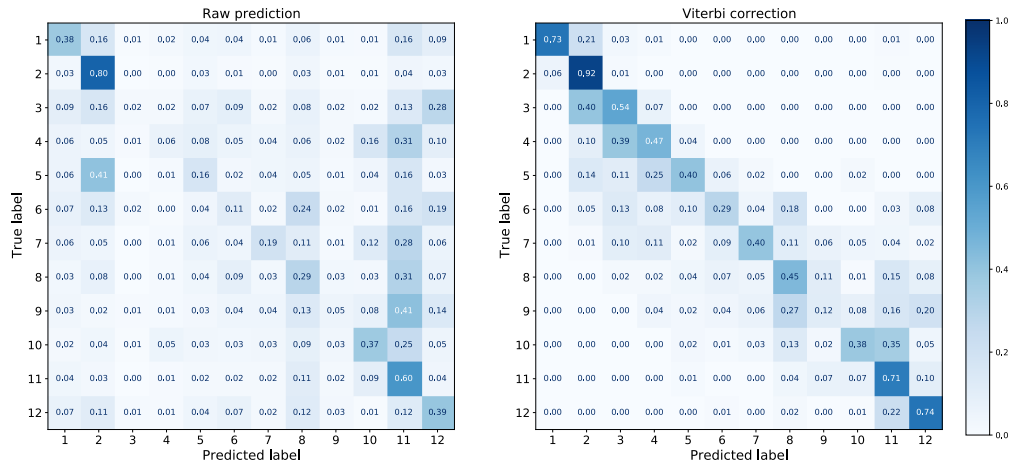
Figure 12: Frame-level confusion matrix on the wound care task. (left) raw HAR (right) after the Viterbi correction.
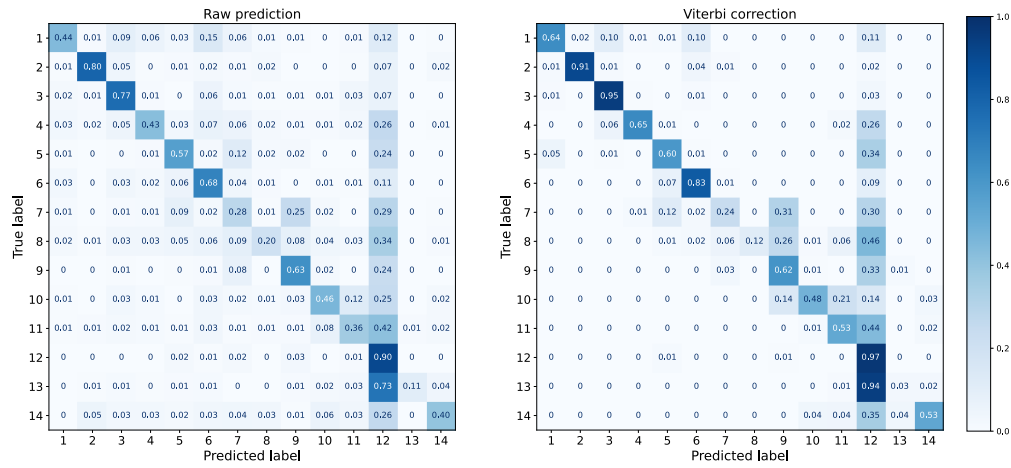

Figure 13: Frame-level confusion matrix on the cooking task. (left) raw HAR (right) after the Viterbi correction.
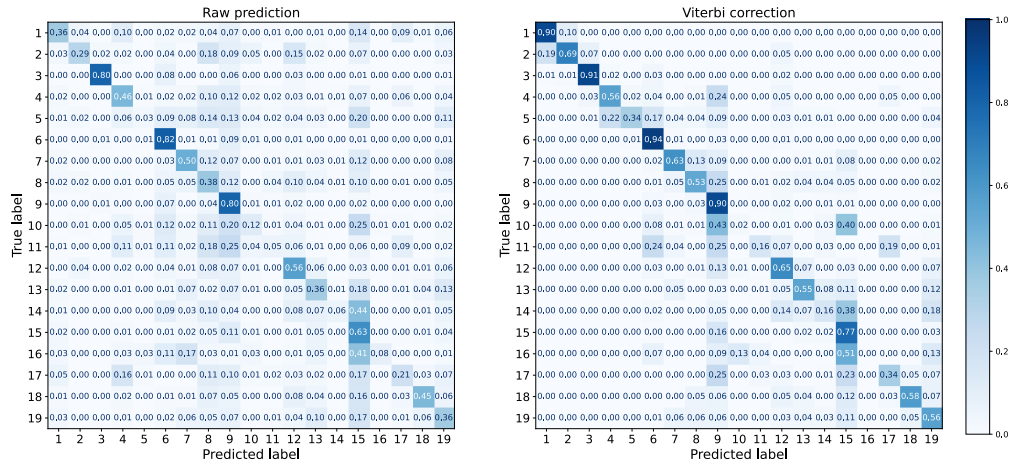

Figure 14: Frame-level confusion matrix on the latte-making task. (left) raw HAR (right) after the Viterbi correction.