AdapTutAR: An Adaptive Tutoring System for Machine Tasks in Augmented Reality

Gaoping Huang*
School of Electrical & Computer Engineering, Purdue University
West Lafayette, Indiana, USA
huang679@purdue.edu

Xun Qian*
School of Mechanical Engineering, Purdue University
West Lafayette, Indiana, USA
qian85@purdue.edu

Tianyi Wang
School of Mechanical Engineering, Purdue University
West Lafayette, Indiana, USA
wang3259@purdue.edu

Fagun Patel
School of Mechanical Engineering, Purdue University
West Lafayette, Indiana, USA
patel648@purdue.edu

Maitreya Sreeram
School of Industrial Engineering, Purdue University
West Lafayette, Indiana, USA
msreeram@purdue.edu

Yuanzhi Cao
School of Mechanical Engineering, Purdue University
West Lafayette, Indiana, USA
cao158@purdue.edu

Karthik Ramani
School of Mechanical Engineering, Purdue University
West Lafayette, Indiana, USA
ramani@purdue.edu

Alexander J. Quinn
School of Electrical & Computer Engineering, Purdue University
West Lafayette, Indiana, USA
aq@purdue.edu

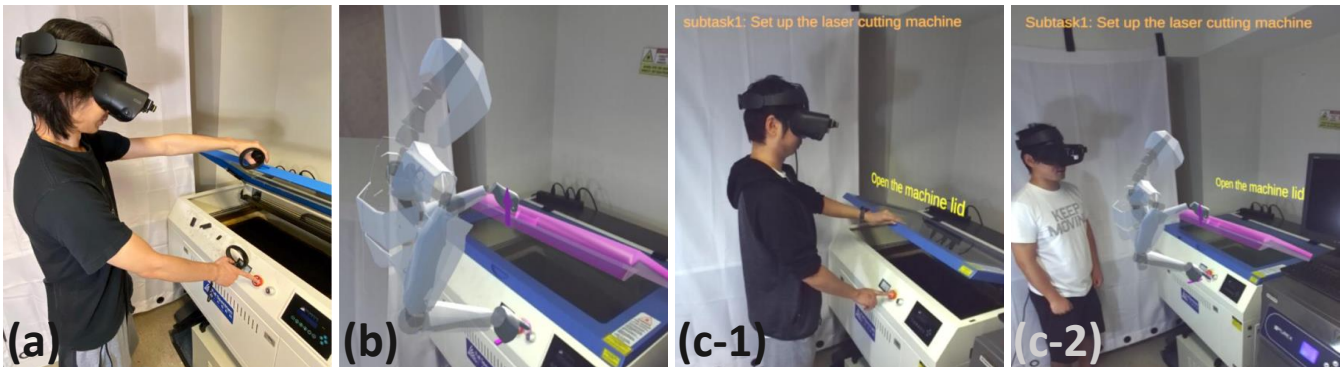


Figure 1: An overview of AdapTutAR workflow. (a) An expert records a tutorial. (b) The tutorial is represented as an avatar and animated components with arrows. The expert can edit the tutorial by adding *subtask* description and *expectation of step*. c) The same tutorial is adaptively shown to two learners. The learner in (c-1) is given less tutoring contents than the learner in (c-2) due to the difference of their experience and learning progress.

ABSTRACT

Modern manufacturing processes are in a state of flux, as they adapt to increasing demand for flexible and self-configuring production. This poses challenges for training workers to rapidly master new machine operations and processes, i.e. *machine task*s. Conventional in-person training is effective but requires time and effort of experts for each worker trained and not scalable. Recorded tutorials, such as video-based or augmented reality (AR), permit more efficient scaling. However, unlike in-person tutoring, existing recorded tutorials lack the ability to adapt to workers' diverse experiences and learning behaviors. We present AdapTutAR, an adaptive task tutoring system that enables experts to record *machine task* tutorials via embodied demonstration and train learners with different AR tutoring contents adapting to each user's characteristics. The adaptation is achieved by continually monitoring learners' tutorial-following status and adjusting the tutoring content on-the-fly and in-situ. The results of our user study evaluation have demonstrated that our adaptive system is more effective and preferable than the non-adaptive one.

*Both authors contributed equally to this research.

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

ACM ISBN 978-1-4503-8096-6/21/05…$15.00
https://doi.org/10.1145/3411764.3445283

CCS CONCEPTS

• **Human-centered computing → Mixed / augmented reality**.

KEYWORDS

training/learning, manufacturing, user state recognition, adaptation

ACM Reference Format:
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 *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 15 pages. https://doi.org/10.1145/3411764.3445283

1 INTRODUCTION

Human workers are the most flexible part of the production process [64]. In the ongoing trend known as *Industry 4.0* [19], workers are expected to operate diverse machinery and other equipment in constantly changing working environments [37]. To meet these challenges, workers must rapidly master the machine operating procedures, referred as ***machine task***s. Numerous tutoring systems have been developed to facilitate the training [7, 9, 20, 49, 58]. These novel tutoring systems show potential to eventually eliminate real-human one-on-one tutoring, which will greatly lower the training cost and increase the scalability of workforce training.

Recorded tutorials permit more efficient scaling than live tutoring which requires in-person training. Prior studies [29, 39, 63] have compared tutoring effects between one-on-one live training and recorded tutorial-based training. Their results indicate that tutorial-based training is effective in efficient remote distribution and scalability, however, traditional one-on-one training has significant better training outcomes. This is because unlike a recorded tutorial which is mostly fixed and static once created, a live tutor can adapt to learners uncertainly during the training and adjust the tutoring content to achieve better results.

This concept of adaptation is particularly important in the *machine task* tutoring scenario, since workers are expected to be more versatile with various machine operations and processes, and the *machine task* environments are highly dynamic and spatial. Furthermore, each worker has their own different innate capability and strengths/weaknesses. In order to achieve better *machine task* skill transfer, it is crucial to design tutoring systems with capability of adapting to the ever changing working environment, as well as each individual worker.

In terms of tutoring presence, prior works have demonstrated the strength of humanoid avatar as a virtual representation of the user [7], for enhancing his/her bodily-expressive human-human communication. Besides, augmented reality naturally supports spatially and contextually aware instructions for interacting with the physical environment. Researchers have shown promises to use AR avatar as a virtual media for *machine task* tutoring applications [7, 57]. On the other hand, annotations [26, 65, 66] and animated components [2, 17, 30] have been widely used in prior AR research to provide tutoring content and guide users.

To this end, we present AdapTutAR, a *machine task* tutorial system with four kinds of AR elements that focuses on *adaptation*. Our system achieves adaptation by actively monitoring both the machine state as well as the user state during the tutoring process. We leverage the benefits of AR in spatial and contextual content visualization, and deep learning in object recognition as well as user activity recognition. The key contributions of this paper are as follows:

- The design of the adaptation model that focuses on spatial and bodily visual presence for *machine task* tutoring.
- The design of corresponding features that enable adaptive tutoring in the recorded-tutorial environment based on *machine task* state and user activity recognition.
- The system implementation that achieves AR avatar tutorial recording, adaptive visualization and state recognition, and evaluation results from our user study

2 RELATED WORK

This section discusses prior approaches to adaptive tutoring for general contexts, and the sources and targets of adaptation for AR/VR specific tutoring systems. These works inform our system design and features.

2.1 General Strategies for Adaptive Tutoring

In human-based tutoring systems, instructors perform a multi-dimensional role, from providing classroom instructions, providing feedback to trainees (questioning, suggesting hints or direct orders) and even varying the difficulty of the training to suit the trainee [54]. Adaptive instructional systems aim to replicate these roles in the absence of a human tutor. These computer-based systems guide learning experiences by tailoring instruction and recommendations based on the goals, needs and preferences of individual learners in the context of domain learning objectives [51]. Thus, the goal of adaptive tutoring is not just to facilitate, but optimize the learning, retention and transfer of skills for users between the training and real-world environment.

Instructional strategies for adaptive tutoring can be grouped into two general approaches: macro-adaptive and micro-adaptive [16, 35]. Macro-adaptive approaches provide adaptation based on metrics collected prior to training. Generally, they use metrics such as learner preference and experience to establish methods for individualized task selection or content difficulty [60]. Micro-adaptive use real-time metrics to provide adaptations in a dynamic fashion. They perform adaptations during the training using factors such as user performance, behavior and errors to provide guidance and feedback [12, 18]. However, to determine which adaptive approach should be used, it is useful to understand the possible sources and targets of adaptation.

Sources of adaptation pertain to factors which cause or trigger the adaptation to occur. They largely stem from learner-based metrics such as individual performance, working memory capacity [38], prior expertise [50], learning preference and traits [18]. Intelligent tutoring systems usually employ a learner model [10, 41] which collects learner data to ascertain their knowledge state, recognize errors and generate adaptations. ***Targets*** of adaptation represent those instructional components which actively change based on the source of adaptation. Broadly, these targets can include the feedback, visual representation of information, sequence of workflow, learning pace and others [18]. Effective and personalized feedback

is important for learning due to the different characteristics of users, as shown in the works by Gutierrez and Atkinson [21] and Bimba et al.[4]. Alternatively, Brusilovsky and Su [5] explored the relation between adaptive visualization and learner knowledge levels, while Beyyoudh et al. [3] focused on providing the optimal sequence of pedagogical steps.

The sources and targets of adaptation define the general adaptive approach of the tutoring. While AdapTutAR uses a combination of macro and micro-adaptive approaches for adaptation, the following sections discuss the different sources and targets of adaptation specific to the AR/VR context.

2.2 Source of Adaptation in AR/VR

AR/VR applications have shown benefits to communication and learning by displaying effective and adaptive information that enhances users' understanding of subjects. While adaptation within the AR/VR context has not been studied extensively, some sources of adaptation in AR/VR tutoring systems can include learner performance, expertise, behavior (gaze, distraction, emotion), task-type and spatial location. From the previous section it can be seen that generic tutoring systems largely monitor user-based factors. In the case of AR/VR, these factors can be classified into two groups: User and Environment [32].

User refers to the person using the application. User's performance is widely used in directing the tutoring workflow, initiating appropriate feedback [31, 54, 62], or in selecting macro and micro-adaptive strategies [52]. Fender et al. leveraged user behavior and object position to adapt the position [14] and size [15] of AR displays. Learner gaze can be used as a measure of transparency [56], allowing the tutoring agent to make inferences about the learner confidence, whether or not guidance is necessary and what they are likely to do next. [50]. Finally, Rodenburg et al. elaborate on the correlation between learner expertise and level of fidelity in simulation based tutoring environments [47].

Environment refers to the physical context where users are interacting. When dealing with machine environments, Cao et al. [7] in their exploratory study categorize the steps of *machine task*s into three types depending on the physical actions performed: *local, spatial* and *body-coordinated* tasks. Their user study suggests that users prefer different visual abstractions of the AR avatar tutors depending on the task type. Lages and Bowman [32] used information about physical surroundings and relative positions of the environment layout to focus on position-adaptation. Additionally, Herbert et al. [26] use spatial 3-D information from the real-world to detect errors, provide feedback and sequence tasks.

By taking information from the user and environment into account, AdapTutAR generates adaptive tutorials for *machine task*s. The next section describes the various targets of adaptation in AR/VR for *machine task*s.

2.3 Target of Adaptation in AR/VR for *Machine Task*s

*Machine task*s can be defined as a sequence of physical and spatial operations involving machines in a production environment [7]. Some examples of AR-based tutoring for machine environments include usage of industrial machinery [8, 46, 68], facility monitoring [34, 67] and vehicle maintenance [11, 24]. Considering that most *machine task* operations involve human motion, the targets of adaptation must focus on using the right type of visualization content for the tutoring and the level of detail.

AR/VR visualizations include the usage of annotations, animated components and virtual avatars. While annotations [26, 65, 66] and animated components [2, 17, 25, 30] have been used extensively in prior AR research to adapt feedback and guide users, avatars have been used to provide effective feedback for learning tasks that primarily require human motion. For example, Tai-Chi training platforms where learners learn from virtual coaches have been researched extensively [22, 42]. Cao et al. suggested the use of avatar as an additional instructional mode [7] after exploring the presence of avatar for tutoring *machine task*s. Similarly, Piumsomboon et al. studied the presence of an adaptive avatar to facilitate the collaboration between a local AR user and a remote VR user [43, 44]. Recently, Loki [57], a bi-directional mixed-reality telepresence system for teaching physical tasks, used the avatar to represent status of the learner and instructor in different physical spaces. By offering customized feedback from the avatar, users gain deeper understanding within synchronous learning.

Level of Detail (LoD) relates to the questions of *when,* and *how* much information should be presented to the user for optimal tutoring. Lindlbauer et al. report an optimization approach leveraging cognitive load and the task environment to adapt MR interfaces to fit the user's context. [36]. Wegerich and Rötting [61] outline a context-aware adaptation system for spatial AR with the goal of displaying unambiguous information at the right time to the user, based on user attributes such as position and perception. For AR browsers, Tatzgern et al. [55] presented an adaptive information density display which used a level-of-detail structure to balance information against potential clutter on the display.

Prior works use these targets of adaptation to provide users with optimal amount of information for tutoring. The different sources of adaptation reviewed in the previous section are linked to various targets of adaptation in our work. AdapTutAR significantly expands on these targets and presents an adaptation model that targets the optimal level of detail based on a combination of user and environmental sources.

3 AR TUTORING ELEMENTS

In our work, we mainly focus on the tutoring of *machine task*s [7], in which a production process involves a compound sequence of *local*, *spatial*, and *body-coordinated* human-machine interactions [27, 53].

AdapTutAR aims to transfer the general process of *machine task*s to workers, such as what component (e.g., knob, lever) to operate, in what order, the exact state to change, and the expected outcome on each operation. Based on the prior work of AR visualization and the nature of *machine task*s, AdapTutAR chooses four types of tutoring elements to convey such sequential and logical knowledge to a learner (Figure 2).

(1) **Avatar**. Since *machine task*s often involve spatial and body-coordinated human-machine interactions, the presence of AR avatar has shown benefits in *machine task* tutoring by improving learners' spatial attention and understanding of

potential movement [7]. Specifically, the humanoid avatar can demonstrate the location of the interaction, the navigation path, and the body pose/gestures to accomplish a step.

(2) **Animated component and arrow**. Given that each step of a *machine task* involves manipulating one or more machine components–such as knobs, buttons [28]–AdapTutAR animates the virtual representation of these interactable component(s). Nonetheless, when the animation is repeated in a loop, users may feel confused about the actual direction of the animation (e.g., clockwise or counter-clockwise). Hence an arrow is added to clarify the direction. The animation and arrow help indicate how the component will look like when it is manipulated by a user.

(3) ***Expectation of step***. When it comes to steps that require a user to set the machine to a specific state or parameter, it is often inadequate to convey the expected value by purely using animated components or arrows. To complement that, AdapTutAR shows expectation (e.g., the yellow text in Figure 2) right next to the animated component to indicate the expected value, such as "Set the printer head temperature to 500 F". The *expectation of step* has more formats than text. For some steps, AdapTutAR shows a virtual model as the expected value, such as a virtual car to be 3D printed or a tool to be used.

(4) ***Subtask* description**. A *machine task* consists of multiple steps. Some consecutive steps may represent a cohesive subgoal, which is called a *subtask*. For example, a *subtask* "Replace the 3D printer head" involves loosing the safety lock, removing the existing printer head, picking up the expected one, installing it, and tightening the safety lock. A *subtask* description is shown at the top-left corner of a user's view to help the user build a higher-level understanding of the *machine task*.

4 FORMATIVE STUDY ON AR VISUALIZATION

To inform the design of the adaptation model, we aimed to understand the performance of a user while learning a *machine task* using AR tutorials. A key objective was to elicit the users' preferences of exploiting the four tutoring elements and the requirements to the tutorial throughout the learning process.

4.1 Participants and Procedure

We recruited six participants (5 male, 1 female) aged 23 to 30. Four participants had experience with AR/VR systems while two did not. No participants had used AR/VR based tutoring systems before attending this study. (Participant: P)

We designed a laser cutting *machine task* consisting of interactions with physical interfaces and spatial navigation within AR environment. The participants were asked to learn the task using a pre-authored AR tutorial where all four tutoring elements were available in each step and the participants could manually toggle on/off any of them and browse along the steps using Oculus hand controllers [1]. Meanwhile, the participants were informed to learn the task in any way they felt efficient and comfortable by utilizing

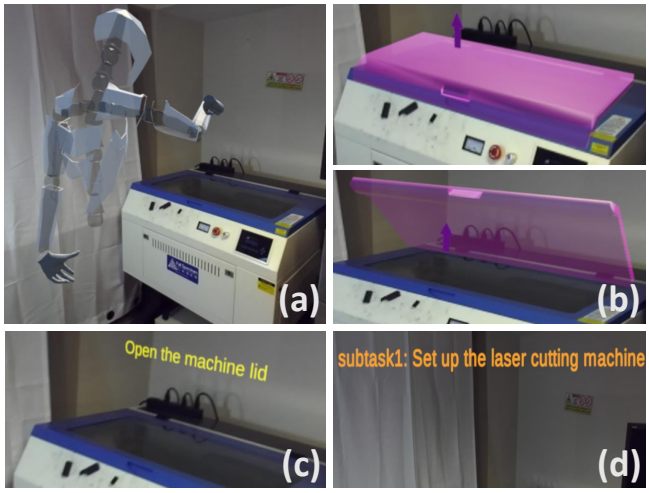


Figure 2: Tutoring elements: (a) Avatar, (b) Animated component and arrow, (c) *Expectation of step*, and (d) *Subtask* description.

the tutoring elements, and the final goal was to remember and conduct the whole task without external assistance. The learning might go over for several times and end when a participant told the researcher he/she had mastered the whole task. The first-person view of the participants was screen-recorded and after the learning period, we interviewed the participants regarding the learning experience and our observations on their performance.

4.2 Findings

We analyzed the participant records in terms of the overall performance and the detailed actions in order to reveal the users' preference to the *machine task* learning. We analyzed 1) the pattern of step changing and tutorial following, 2) the timings when the participants toggled on/off the tutoring elements, 3) the combination of the tutoring elements at each step, and 4) the choices above at different learning stages. In addition, we analyzed the participants' bodily performance including their standing location, attention allocation, and so on. Finally, we distilled the higher-level design goals from our observations and the participant feedback.

4.2.1 Overall Learning Flow (F1). Although the participants were able to navigate back and forth along the whole tutorial using the hand controllers, all six participants learned the task by following the tutorial step by step and repeated the whole task for multiple times. *"I think the order of these steps is critical to understanding the whole task. So, instead of mechanically remembering every single step, I learned them as an integrated story." (P6)* Moreover, all participants would only progress to the next step after ensuring the current step was completed correctly. *"I'd feel more confident if the system could tell me whether I did it correctly." (P3)* The performance and responses highlight that the adaptive tutoring model should be able to recognize the correctness of an operation in real-time and actively lead him/her to move forward in the task to ensure a fluent learning flow.

4.2.2 Combination of the Tutoring Elements (F2). Overall, all participants agreed that the provided four tutoring elements were useful and sufficient for the learning. Yet, at different stages of the learning process, the participants chose different combinations of the tutoring elements. All six participants kept all four elements in the first trial, and went through every element in each step. *"The avatar was important when I first learned the task because it told me where should I focus on. And I also read the subtask description to briefly understand the purpose of the task." (P5)* All participants agreed that as they were more practiced, the required tutoring elements shifted from specific demonstration to high-level description. *"After I knew those operations, the avatar was not that useful, but distracted me. So I turned it off." (P2)* These findings revealed that at different learning stages, the importance of each tutoring element varies. So the system should dynamically change the displaying tutoring elements as the learning progressed.

4.2.3 When to Show/Hide Tutoring Elements (F3). We asked the participants to only keep the necessary tutoring elements while learning. First, we noticed that all toggle-off actions happened at the beginning of a step when a participant was clear he/she could master it. However, we observed that the toggle-on actions happened in more complicated scenarios. Compared to the toggle-off cases, before toggling on an element, the participants performed additional actions such as walking around the machine, attempting to operate an interface, correcting an operation rapidly and so on. *"Actually I was first trying to look around to find the next target, then if I couldn't, I turned on more." (P1)* This disclosed a need for the adaptation model first to understand the current state of the learner, then either provide more tutoring elements or reduce them. Additionally, we observed that timings when they turned on the tutoring elements varies at different learning progresses. *"When I almost learned everything, if I was stuck, I'd first recall the step, then turn on the elements. But initially, I didn't know much, so directly turned them on." (P4)* It unfolded another requirement for the model to adjust the timing to change tutoring elements accordingly.

4.2.4 Step-dependent Behaviors (F4). For different steps, the participants selected different tutoring elements in the same trial. Meanwhile, when repeating the same step in different trials, and doing similar steps in the same trial, we observed that all participants gradually reduced the tutoring elements. *"The steps were different in some cases, so I'd like to use different elements. But there were some similar ones, maybe the system could show me the previous choice." (P1)* Meanwhile, when a step required spatial movements or complicated body-coordinated actions, the participants would spend more time for learning. *"Some steps were harder to learn, so I needed more time before I could turn off some tutoring elements." (P3)* Inspired by these findings, the model should also consider the nature of each step and the transition between any two steps.

5 ADAPTATION MODEL

To develop a tutoring system that can dynamically adapt the tutoring elements to match what a user actually needs, we organized the tutoring elements into five levels of details (LoDs), and further developed an adaptation model to adjust the LoDs in real-time (Figure 3). The key model includes four phases. Firstly, the system presents the tutorial at a given LoD. Secondly, the system collects the inputs from a user and environment in real time. Thirdly, the system performs low-level state recognition that recognizes the machine state, the user's basic mode, and region of interest (ROI). Finally, the system uses the low-level state to estimate the user's higher-level state, such as being stuck or not. Such estimation is transferred back to adapt the LoDs in the first phase.

5.1 Features to Adapt: Level Of Detail

The formative study confirms that each tutoring element serves a different role in conveying information, and further indicates that their necessity varies at different stages of the learning process *(F2)*. Therefore, we organize the tutoring elements into five levels of details (LoDs) as below.

- **LoD 5**: show all four tutoring elements.
- **LoD 4**: exclude avatar from LoD 5.
- **LoD 3**: exclude animated components and arrows from LoD 4. This essentially means it only shows expectation and *subtask* description.
- **LoD 2**: exclude step expectation from LoD 3, namely, just showing *subtask* description.
- **LoD 1**: show nothing.

When the LoD decreases, the tutoring elements are hiding gradually. The difficulty increases since there are fewer hints. In particular, for LoD 1 and 2, learners do not get direct hints about what component to operate nor what state to set to, which forces them to recall the details instead of being informed directly. On the other hand, learners who have gone through the same operations for multiple times may not need the detailed information provided in high LoDs.

As the first phase of each step, the system loads the historical LoD to determine what tutoring elements to present. If there is no historical data, the system shows the step with a default LoD (5).

5.2 Sensing Input

In this phase, AdapTutAR keeps collecting two categories of information: user and environment. User information includes the position and orientation of the AR headset as well as the first-person view of the user. Environment information includes the positions, orientations, and dimensions of the animated components and avatar.

5.3 Low-level State Recognition

In this phase, the inputs are used to recognize machine component state and user's basic state, which is further used to perform the higher-level state recognition in next phase.

5.3.1 Recognizing Machine Component State. The goal of machine state recognition is to detect what state the user has set the physical component to. This is required because the physical machine may not have sensor itself that could report the current state. In order to control the playback of the tutorial, its state change must be detected.

Prior works mostly focused on object detection and recognition (such as [45]), while a few focused on recognizing the specific states

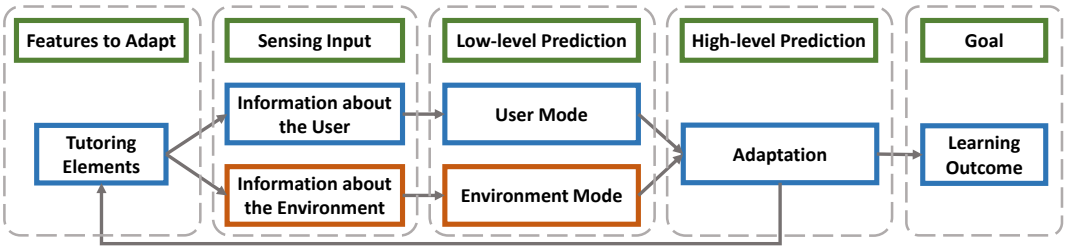


Figure 3: The Adaptation Model. Green boxes indicate the phases of adaptation.

of an object [13]. However, these methods cannot be directly applied to our case where multiple identical objects may be visible on the same machine interface. For example, two knobs and two buttons are visible in Figure 5. The challenge is that after recognizing the states of these objects, the system needs to know which state belongs to which object. Inspired by LabelAR [33], AdapTutAR leverages the AR components that are aligned to the physical components. Besides the primary role of giving animations as instructions, AR components serve an additional role in providing the positions and dimensions in the world space, then AdapTutAR can obtain their 3D bounding boxes and further compute the 2D bounding boxes on the screen. Such bounding boxes help identify an object uniquely even when there are multiple identical objects within the scene. Finally, a Convolutional Neural Network (CNN) model is trained to recognize their object states within each bounding box. The detail is discussed in the Implementation section.

5.3.2 Recognizing User's Basic Mode. The goal of user state recognition is to identify what *basic mode* the user is in, including *static observation*, *navigation*, and *interaction*. To classify *interaction* mode, the key is to know whether a user touches the physical component or not. An approach similar to the aforementioned machine state recognition is used. For all visible machine components, the system crops out the camera images based on their bounding boxes, groups them into a batch, and predicts hand touching in parallel. If any component is touched by the user, the mode is classified as interaction. The user's state of *static observation* or *navigation* is predicted using the AR headset's position and orientation using a pretrained Support Vector Machine (SVM) model.

5.3.3 Classifying Region of Interest (ROI). To classify whether a user is looking within ROI or outside of ROI, we first get the location of the target object(s) and avatar for a particular step, and compute whether they are visible by the user. This is essentially checking whether any of these objects is within the field of view (FOV) of the AR headset. If none of them is within FOV, the system classifies the user as looking outside of ROI; otherwise, within ROI.

5.4 Higher-level State Recognition and Adaptation

In this phase, the system estimates the user's state by forming a finite state machine and using the low-level recognition as inputs to drive the state transition.

5.4.1 Scenarios and States. We leverage the findings from the formative study *(F3)* and identify four scenarios in which users decided to turn on more tutoring elements, including (S1) unaware of the target, (S2) unaware of the operation, (S3) interact with wrong interface, and (S4) interact with the correct object for too long without setting to the expected state. The core of adaptation is to estimate whether a user is currently under one of four scenarios and thus needs more tutoring content. To that end, we develop a finite state machine that takes the lower-level state recognition as input to help infer the states of a user (Figure 4). The four scenarios correspond to the exceptions of four higher-level states, including "Viewing outside of ROI", "Viewing within ROI", "Manipulating wrong object", and "Manipulating correct object". Due to their high correspondence, we also denote the four states as S1-S4, respectively. By monitoring how long a user stays in each state, the system determines whether the user enters one of the four scenarios.

5.4.2 State Transition. At the beginning of a step, the user immediately transits into one of three states: "Viewing outside of ROI", "Viewing within ROI", and "Changing perspective". The first state requires that the user is in static mode and looking outside of ROI, while the second state requires that the user is in static mode and looking within ROI. When the user is in navigation mode (e.g., walk or move head), their state will be transited to "Changing perspective". Once the user pauses walking or moving head, the state will be transit into the first or second state accordingly.

When the user touches a machine component, the user state transits into one of two states: "Manipulating wrong object" and "Manipulating correct object". This transition depends on whether or not the touched component is the expected one in the current step. While the user is manipulating the correct object, our system keeps recognizing the machine state and comparing it to the expected one. If matched, the current step is done. If not and the user stops manipulating, the user state transits back to one of three states related to viewing and changing perspective.

Each state has an independent timer which resets at the beginning of each step. When a user transits from one state to another, the timer of the original state pauses while the timer of the new state starts ticking. If a user remains in one state for too long (i.e., accumulated time > threshold), the system estimates that a user may be stuck in one of four scenarios. For example, if a user stays in "Viewing outside of ROI" state for time longer than $threshold_1$, the system estimates that the user is "unaware of the target" *(S1)*. Likewise, if a user stays in "Viewing within ROI" state for time longer than $threshold_2$, the user is inferred to be "unaware of the operation" *(S2)*. Consequently, an event is triggered to increase the level of detail (LoD) and reset all timers. The calculation of thresholds can be found in section 5.4.3.

Finally, if a user completes a step, the LoD is decreased by 1 *(F3)* and saved into user's profile for future reference. Note that the

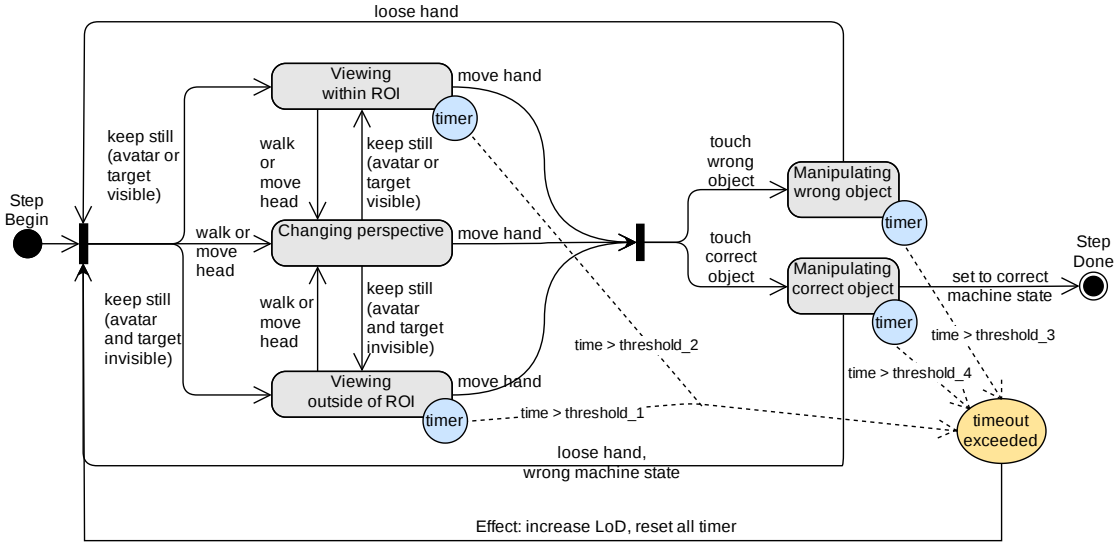


Figure 4: Inferred state of a user in a single step via Finite State Machine.

LoD data is tied to a component, not to a step. This is because a component may be involved in multiple steps of the same tutorial or be shared in different tutorials that involve the same machine. Therefore, binding LoD data to a component rather than a step supports better reuse of the user's learning record. As mentioned in the first phase earlier, the system loads the historical LoD at the beginning of each step. Here, it only loads the most recent LoD of the component, and ignores the older LoD record(s) if any.

In summary, the adaptation model leverages both the information from the historical record and the real-time inputs, which is a combination of macro and micro-adaptive approaches [16, 35].

5.4.3 Timer Threshold. The key is to find a proper threshold for each state/scenario. The first empirical observation is that the threshold should refer to the actual time spent by the expert in each step. A more complicated step takes longer time than an easy step so that the former should have a larger threshold than the latter. Based on section 5.3.2, the total time spent in step i can be further decomposed into the time spent in observation ($observeTime_i$), navigation ($navigateTime_i$), and interaction ($interactTime_i$). Moreover, the threshold of each state should refer to the most relevant type of time. For example, the states related to manipulating objects (S3 and S4) should refer to the interaction time, while the states related to viewing (S1 and S2) should refer to observation and navigation time. Therefore, let $ReferenceTime_i$ denote the reference time of states (S1-S4) in step i:

$$\mathbf{ReferenceTime}_i = \begin{bmatrix} t_i^{s1} \\ t_i^{s2} \\ t_i^{s3} \\ t_i^{s4} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0.5 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} observeTime_i \\ navigateTime_i \\ interactTime_i \end{bmatrix} \quad (1)$$

For example, the reference time of manipulating correct object (S4) in step i is $t_i^{s4} = interactTime_i$. Also, when a learner is manipulating a wrong object (S3), it is supposed to have a smaller threshold so that the hints can be shown faster. Therefore, an empirical value (0.5) is chosen so that $t_i^{s3} = 0.5 \times interactTime_i$.

Lastly, the current LoD matters. A larger LoD implies that a user is less proficient in this step so that the system should tolerate a larger threshold. Let $Threshold_i$ denote the thresholds for states S1-S4 in step i:

$$\mathbf{Threshold}_i = \begin{bmatrix} threshold_i^{s1} \\ threshold_i^{s2} \\ threshold_i^{s3} \\ threshold_i^{s4} \end{bmatrix} = f(LoD_i) \times \mathbf{ReferenceTime}_i \quad (2)$$

where $f(x)$ is a factor that scales the reference time based on the current LoD. In this project, we take:

$$f(LoD) = 1 + log_5(LoD) \quad (3)$$

When LoD=5, $f(x)$ is 2, which means the threshold is twice of the reference time. This is because at LoD 5, a learner is inferred as a novice so that they may spend $1 \times referenceTime$ in purely watching the tutorial and $1 \times referenceTime$ in following the tutorial. On the other hand, when LoD=1, $f(x)$ is 1 because the system infers the user as proficient to this step and tolerates the same time as the reference time. Using a *log* function rather than a linear function is to make the threshold decrease slower in large LoD (4 and 5) and faster in small LoD (1 and 2).

6 ADAPTIVE TUTORING SYSTEM

To support effective apprenticeship for *machine task*s in workshops or factories, we designed and implemented AdapTutAR. AdapTutAR is an AR-based authoring and tutoring system that enables an expert to record a tutorial that can be learned by different workers in an adaptive way.

6.1 Workflow Illustration (Overview)

AdapTutAR consists of three modes: 1) Authoring Mode in which an expert can record a tutorial; 2) Edit Mode in which the expert

can edit the tutorial; and 3) Learning Mode in which workers can learn the tutorial.

6.1.1 Prerequisite: Setup Training Environment. Before an expert can record a tutorial for a machine, they need to set up the training environment first. This requires that a machine has a digital copy aligned with the physical counterparts, and also the machine state recognition model has been trained. However, since the focus of this paper is not about setting up the environment but the adaptive tutoring within the environment, for simplicity, we assume the experts are given a machine that has already been set up. Without losing generality, an expert can also use the following process to set up a new training environment. First, the expert uses the hand-held controllers to align the virtual components with their physical counterparts in the AR world. Secondly, they can follow the pipeline mentioned in the later subsubsection 6.2.2 to collect a dataset for machine state recognition. Once enough data is collected, our system can fine-tune the CNN model with the dataset. The dataset only need to be collected once for one type of machine.

6.1.2 Authoring Mode. Tutorials are authored using an natural embodied movement, where the system records the expert's body motion by tracking the position and orientation of the AR headset and two hand-held controllers (Figure 1a). In addition, the expert can manipulate the virtual component(s) through different gestures of virtual hands powered by the controllers. During the operating process, the human motion and the 6 DoF poses of the virtual components are recorded. The recorded human motion will be represented as avatars while the recorded pose sequences of the virtual components will become AR animations. To partition the entire tutorial into steps, the expert needs to explicitly starts and stops recording each step by pressing the joystick on the controllers.

6.1.3 Editing Mode. Once all steps are recorded, the expert can enter the Edit Mode to label descriptions. The expert can pick a step to add expectation or select several consecutive steps to add a *subtask* description. To add a subtask description, the expert can enter a short sentence via virtual keyboard. To add an expectation, the expert first creates the text in a similar way and then uses controllers to anchor it to the proper position in the environment.

6.1.4 Learning Mode. A learner wears the AR headset and starts to follow the first step of the tutorial without hand-held controllers (Figure 1c). Four types of tutoring elements may be shown/hidden dependent on the current LoD. By default, a user starts with LoD 5 so that they are given all elements to guide how to operate. As a learner may need to repeat the tutorial for multiple trials before comprehending it, the system keeps adapting the tutoring content for each step based on their historical learning progress and the current behavior. This is achieved by the aforementioned Adaptation Model that is running in the background. It also monitors if the learner has set the component to the expected state. The tutoring elements remains until the learner operates correctly *(F1)*.

6.2 Implementation

6.2.1 System Hardware and Software Setup. The see-through AR platform is built by attaching a stereo camera (ZED Dual 4MP, 720p) in front of a VR headset (Oculus Rift [1]). Four external Oculus IR-LED sensors are used to track the human body motion with an effective area of 3 x 3m. Two Oculus Touch Controllers enable authoring by an expert. The main AR interfaces are developed with Unity3D [59], and the predictions are made with a backend server running aiohttp[1] web framework in Python. The backend server loads the models trained by Tensorflow (v2.1) and SVM. Both Unity3D and backend server run on the same PC (Intel Core i7-9700K 3.60GHz CPU, 32GB RAM, NVIDIA GeForce RTX 2070). The stereo camera provides built-in streaming functionality that can be accessed by the server. Unity3D sends data to the server via Socket.IO, including the objects to be tracked, their bounding boxes, and the positional data of the headset. In return, the server sends the predicted machine state and user state back to Unity3D via Socket.IO.

6.2.2 Recognizing Machine Component State. As mentioned earlier, we leverage the bounding boxes of virtual components to uniquely identify physical components. As there are different types of machine components, we developed an efficient pipeline to collect dataset based on video streaming and bounding boxes. Note that if some components are identical, users only need to collect dataset based on their type (e.g., knob, lever), rather than individual components. First, a user sets a physical component to a specific state or sets multiple components to specific states, such as "1" for knob in Figure 5. Then the user selects their virtual counterpart(s) in our system, sets the state(s) to match the physical one(s), and starts video streaming of ZED camera. The video stream is automatically cropped into RGB-D images based on the bounding boxes and also labelled with the current states and types. To make the dataset comprehensive, the user needs to look at the object(s) from various heights, places, and angles. Such process can be repeated to cover the remaining states of the component(s) as well as other interactable components of the machine.

The collected images are used to train a CNN model for state recognition, as shown in Figure 5. First, each image is resized to 100x100x3. Then data augmentation is done by adding random hue (max_delta=.2), saturation (0.1~2.0), contrast (0.3~1.0), and brightness (max_delta=.5). The feature extraction of the CNN model is based on MobileNetV2 [48]. After feature extraction, the output size of the base model is 4x4x1280, which follows by layers of MaxPooling2D, Flatten, two fully connected layers (units=1024 and 64) with Relu activation, Dropout(0.5), and a fully connected layer (units=number_of_component_types) with "softmax" as activation. The loss function is "tf.keras.losses.SparseCategoricalCrossentropy" and the optimizer is SGD.

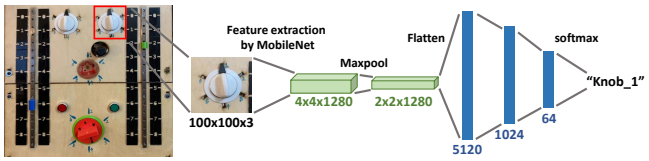


Figure 5: CNN model for machine state prediction based on bounding boxes.

[1]https://docs.aiohttp.org/

6.2.3 Recognizing Hand Touching. This is similar to the machine state recognition above. The difference here is that instead of setting the state of a component, a user needs to act two states on each component type, including "hand not touching" and "hand touching". Overall, 130k images were collected for 9 component types by four volunteers. Finally, the images cropped out by the bounding boxes were used to train a different CNN model.

6.2.4 Recognizing Static Observation and Navigation. We used Support Vector Machines (SVMs) to recognize the static observation and navigation, which have demonstrated high performance when applied to human and animal activity recognition tasks [23, 40]. The feature vector is computed by taking the magnitude difference between kth and 0th frames in a window ($k = 0, ..., windowSize$) for each user head position $\mathbb{R}^3$ and orientation $\mathbb{R}^4$ vector. If the absolute magnitude difference is greater than the threshold, features describing changes in the head position and orientation are set to true. Optimal magnitude thresholds were determined by grid search. Three volunteers generated 90 samples for these two states in which each sample lasted about 10-20s. By performing a grid search, our features were extracted using a window size of 1.3s with an overlap of 0.56s (stride).

6.3 Preliminary System Evaluation

AdapTutAR relies on the capabilities of the low-level state recognition. To evaluate these capabilities, we conducted a preliminary system evaluation.

6.3.1 Accuracy of Machine State Recognition. We built a mockup machine with 9 types of machine components to train and test the model. Figure 5 shows one side of the mockup machine. By using the pipeline in section 6.2.2, three volunteers collected 171K images for 9 component types with 31 distinct states. For example, a knob has 6 states, a key hole has two states (inserted or not), and a switch has two states (on or off). Given that the video streaming is 60 FPS and the user keeps changing the view angles, we save one image every 8 frames to avoid identical images. The training took 10 hours on an NVIDIA GeForce RTX 2070. Before the test, each component of the mockup machine was set to a particular state that was entered into the system as ground truth. During the test, the tester wore the AR headset and looked at the component from different angles for approximately 3 seconds. The video stream was cropped out based on the bounding boxes and sent to the trained model directly. Each batch of images took about 0.11s in prediction. Then the predicted states were compared with the ground truth.

Three researchers participated in the test and produced about 2k results for all the 31 states of 9 component types. The overall classification accuracy was 89.1%. Specifically, the levers and knobs produced small errors (95.5% accuracy) while sliders and key holes produced larger errors (85.3% accuracy). In general, the system can satisfy the requirements of machine state recognition.

6.3.2 Accuracy of Hand Touching. A similar approach was used to test the accuracy of hand touching. The difference was that for each component type, the tester's left and right hands alternatively touched the component. Three researchers participated in the test and produced 913 results. The overall classification accuracy was 93.4%, which validated the feasibility of our system in accurately recognizing hand touching on machine interfaces. A typical error happened when the hand was close to, but had not touched the component. Such scenario was often mis-classified as hand touched. Fortunately, such error did not greatly affect the adaptation model because if a user moved the hand close to a component, it implied that the user intended to touch it.

6.3.3 Accuracy of Classifying Static Observation and Navigation. During the test, participants performed three actions: 1) standing still and moving head slowly; 2) standing still and moving head drastically; and 3) walking. The first action should be classified as static observation and the last two actions should be navigation. During the test, three researchers performed each action for roughly 5 seconds, and repeated for 3 times. By splitting the sequences by the window size (1.3s) and stride (0.56s), there were 241 predictions. The overall accuracy was 92.1%. The errors were partly due to the transition from one state to another. This result indicates that the system can detect the user's basic mode accurately.

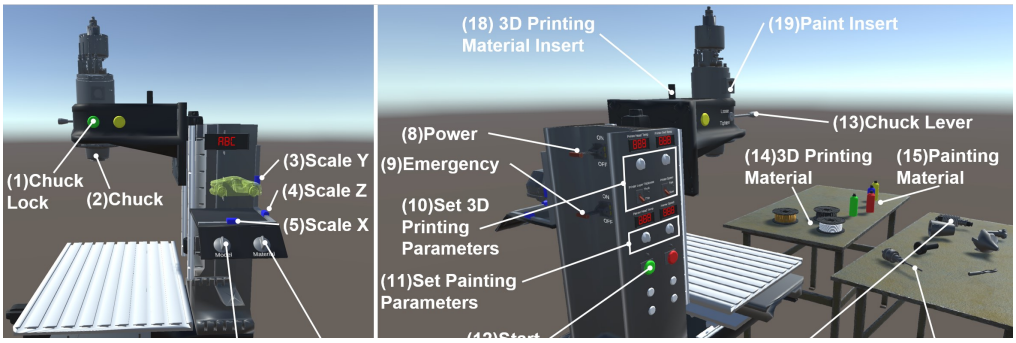
7 USER STUDY

7.1 Study Setup

Complying with the requirements of social distancing for COVID-19, we conducted a 2-session remote user study in Virtual Reality (VR). Since the remote users had no access to the real machines, we built a virtual multi-function machine that enables 3D printing and painting (Figure 6), which was inspired by prior works [6, 36] that validated key features of AR systems in VR. The virtual machine and the two tooling tables were located within a $3m \times 4m$ virtual space. The VR application was sent to the users and the user study was completed using their own VR devices.

During the user study, the users learned a 28-step plastic toy fabrication task using the virtual machine (Table 1). The users had to set the machine parameters using knobs, buttons, switchers or sliders (*local tasks* e.g. step 1 to 5), to deliver correct raw materials (*spatial tasks*, e.g. step 7, 8, 10, 11) and to assemble the tools properly (*body-coordinate tasks*, e.g. step 8, 19, 21). The adjacent steps that served a high-level purpose were grouped into one *subtask*, e.g. the purpose of step 6 to 9 was tooling installation. Some of the steps (e.g. step 8) could be accomplished only if some previous steps (e.g. step 6) were conducted correctly. The users were interacted with the machine using VR hand controllers. Note that in the VR simulation, we directly used the collision between the VR controllers and the machine components to detect the interactions rather than the image classification technique in AR environment. Consequently, the accuracy of machine state and hand touching recognition in VR reaches 100%, which is different from the aforementioned accuracy in AR. We discuss the limitations and mitigation in the next section.

We recruited 24 users (19 male, 5 female, aged 18 to 35) to our remote user study. 19 out of 24 users have engineering background while the other 5 have science background. 11 users have AR/VR experience and 15 users have hands-on machine operation experience. Nine users owned VR devices and shared with their friends or roommates for our user study. Specifically, 14 users used Oculus Rift [1] while 10 users used Oculus Rift S [1]. None of the users had experience with our system before. All the users were compensated with a $20 gift card for the 1.5-hour user study.



Subtask	Step		Widget
Set output model to a Jeep vehicle with PLA material	1	Select model	6
	2	Change material	7
	3	Change scale X	5
	4	Change scale Y	3
	5	Change scale Z	4
Change tool head to printer head	6	Loose chuck lever	13
	7	Pick up printer nozzle	16
	8	Install printer nozzle	1,2,16
	9	Tighten chuck lever	13
Install PLA 3d printing material	10	Pick up printer filament	14

Subtask	Step		Widget
cont.	11	Install printer filament	18
Set parameters for PLA 3D printing material	12	Turn on machine	8
	13	Set head temperature	10
	14	Set bed temperature	10
	15	Set layer thickness	10
	16	Press start button	12
Change tool head to painter head	17	Turn off machine	8
	18	Loose chuck lever	13
	19	Remove printer nozzle	1,2,16
	20	Pick up painter nozzle	17

Subtask	Step		Widget
cont.	21	Install painter nozzle	1,2,17
	22	Tighten chuck lever	13
Install painting material	23	Pick up painting material	15
	24	Install painting material	19
Set parameters for painting	25	Turn on machine	8
	26	Set head temperature	11
	27	Set painting time	11
	28	Press start button	12

Table 1: Tutorial used in the user study that involves 3D printing and painting. Widget # is referring to Figure 6.

7.2 Study Design

We evaluated the benefits and limitations of our adaptation model by comparing an adaptive VR tutorial (noted as *adaptive*) with a similar VR tutorial that had no adaptive features (noted as *non-adaptive*) through a within-subject study. In *adaptive* tutorials, the VR tutorial elements were adjusted following the strategy proposed in section ADAPTATION MODEL, while in *non-adaptive* tutorials, the VR tutorial elements were displayed at a fixed LoD of 5 (none of the tutorials elements were hidden). The users were requested to learn two *machine task*s (Table 1) in two sessions with the two types of AR tutorials respectively.

Both the two *machine task*s were plastic toy fabrication tasks but differed in fabricated models (step 1), materials (step 2, 10), sizes (step 3 to 5), colors (step 11) and machine parameters (step 8 to 10, 26, 27) to avoid the users from remembering the task in the last session. However, both tasks shared similar machine interfaces and step orders. To counter-balance the learning effects, we separated the users into two groups randomly. Specifically, 12 users followed the *adaptive* tutorial in session 1 and the *non-adaptive* tutorial in session 2. In contrast, the other 12 users followed the *non-adaptive* tutorial in session 1 and the *adaptive* tutorial in session 2. The users were not informed with the tutorial conditions.

Each session contains a tutoring section and a testing section. In tutoring section, the users had up to 30 minutes to learn the task by following the tutorial and performing machine interactions. Both of the tutorials were able to proceed to the next step automatically when the user conducted a step correctly. The tutorial repeated from the beginning when a user completed the last step. After the users claimed they had understood and remembered the task, or reached the time limitation, the researcher terminated the tutoring section. Then the users entered the testing section after a 3-minute rest. In the testing section, the users completed the task with all the AR tutoring elements hidden.

In each session, users repeated the tutorial multiple times before they entered the testing section. After session 1, users would change from *novice*, who had little experience in the machine and environment, to *proficient*, who could clearly describe the task purpose and complete the required operations without external hints.

Note that those proficient users were not considered as proficient in all machine tasks but only in the second task, provided that the second task shared similar machine operations and step orders with the first task. Thus, we were able to evaluate the user performance under different conditions, e.g., *novice* with adaptive, *proficient* with adaptive, *novice* with non-adaptive, and so on.

7.3 Data Collection

During each tutoring section, we recorded the total time that the user took and the times that the tutorial had repeated. To better understand the user's behavior, we also recorded the LoD of each step and the reason if the LoD changed (e.g. hesitate for too long, manipulate the wrong object). After the tutoring section, we let the users to evaluate their learning progress using 5-point Likert-type questions (Figure 7 left). For testing section, we used the time that the user consumed as well as the number of mistakes to quantify the learning outcome. After the two sessions, the users voted for their favourite AR tutorial. Further, users rated their subjective feelings about the adaptive features of AdapTutAR using another 5-point Likert-type questions. Finally, A conversational interview was conducted and recorded regarding the the reason why the users preferred an AR tutorial and the insights to improve the adaptive features of AdapTutAR. Additionally, the users' first personal view in VR was recorded for further analysis.

7.4 Results

In this subsection, we present objective performance and the subjective ratings of this study.

7.4.1 Self Rating on Learning Experience. After each tutorial section, the users rated their learning experience of performing the *machine task* using the 5-Point Likert Scale questionnaire (Figure 7). We separated the users based on whether they were novice or proficient and which AR tutorial that they just used. All users reported that they could operate the machine correctly ($M = 4.71, SD = 0.50$) and understand how the machine works ($M = 4.86, SD = 0.37$). Meanwhile, all users were generally confident to finish the task without hints ($M = 4.5, SD = 0.69$), and felt they remembered each step of the task ($M = 4.58, SD = 0.73$). While the proficient users who had used the *adaptive* tutorial rated themselves slightly higher then other users, an one-way ANOVA performed on each group of the ratings showed that there was no significant difference in the ratings regarding "Accuracy" ($p = 0.10$), "Understanding"($p = 0.12$), "Memorization"($p = 0.46$), and "Confidence"($p = 0.27$). In spite of the AR tutorials and the background, all users reported that they had mastered that *machine task*.

7.4.2 Objective Performance. We compared the performance of the novice users and proficient users respectively due to the cognitive gap between the novice and the proficient regarding the *machine task*. A t-Test was performed on each pair of the data. Regarding the learning time, the novice users who used the *adaptive* tutorial took significantly more time ($M = 23.0, SD = 7.3$) in tutoring section than the ones who used the *non-adaptive* tutorial ($M = 16.5, SD = 5.7, p = 0.023$). A similar conclusion could also be drawn from the proficient users (*adaptive* $M = 10.1, SD = 2.9$, *non-adaptive* $M = 7.6, SD = 1.7, p = 0.029$). The novice users who were

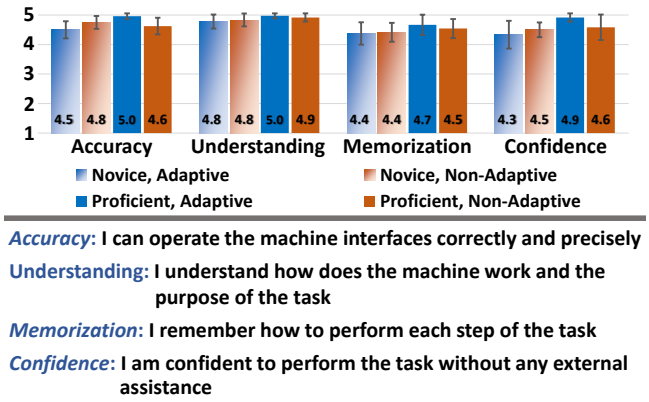


Figure 7: The users' self-evaluation of the learning progress.

using the *adaptive* tutorial required more repeats of the tutorial ($M = 4.1, SD = 1.0$) than the ones with the *non-adaptive* tutorial ($M = 3.3, SD = 1.2, p = 0.017$). The proficient users required similar times of repetition (*adaptive* $M = 2.6, SD = 1.2$, *non-adaptive* $M = 2.3, SD = 0.5, p = 0.50 > 0.05$). Consequently, the users who used the *adaptive* tutorials needed more time (novice $M = 7.2, SD = 2.2$, proficient $M = 3.6, SD = 1.0$) than the user with the *non-adaptive* tutorial to go through the tutorial for one time (novice $M = 4.5, SD = 1.7$, proficient $M = 2.7, SD = 1.0$). In the testing section, novice users using the *adaptive* tutorial took slightly shorter time ($M = 3.2, SD = 0.6$) then the novice users with the *non-adaptive* tutorial ($M = 3.9, SD = 1.3, p = 0.13 > 0.05$), while the proficient users took approximately same time (*adaptive* $M = 2.8, SD = 0.4$, *non-adaptive* $M = 3.2, SD = 0.9$, $p = 0.27 > 0.05$). Notably, the novice users who used the *non-adaptive* tutorial made more mistakes ($M = 2.17, SD = 1.52$) than the ones with the *adaptive* tutorial ($M = 1.00, SD = 1.04, p = 0.039$). The difference between the proficient users was not obvious since they were making few mistakes (*adaptive* $M = 0.41, SD = 0.66$, *non-adaptive* $M = 0.75, SD = 0.75, p = 0.26 > 0.05$). The results are presented in Figure 8.

7.4.3 User Preference Vote. The users voted for their favorite AR tutorial based on their overall experience as well as considering the training efficiency, the learner's understanding of the task, and the comfort of the learning experience respectively (The Figure 9 Left). Overall, most of the users preferred the *adaptive* tutorial (21 out of 24). Meanwhile, the users also agreed that the *adaptive* tutorial delivered a more comfortable learning experience (21 out of 24). In terms of the efficiency and understanding, a little more users (about one-third) chose the *non-adaptive* tutorial, while the majority of the users (about two-thirds) still preferred the *adaptive* tutorial.

7.4.4 Subjective Rating. The Likert-type ratings regarding the adaptive features collected from the user study are shown in Figure 9 right. In general, the users agreed that the *adaptive* tutorial provided appropriate information in time (Q7: $M = 3.8, SD = 0.9$, Q8: $M = 4.8, SD = 0.4$). "*The adaptive tutorial showed the tutorial elements that met my requirements. The AR avatar is the most helpful at first because it was straightforward and intuitive. Later I found the subtask description was more helpful because it reminded me*

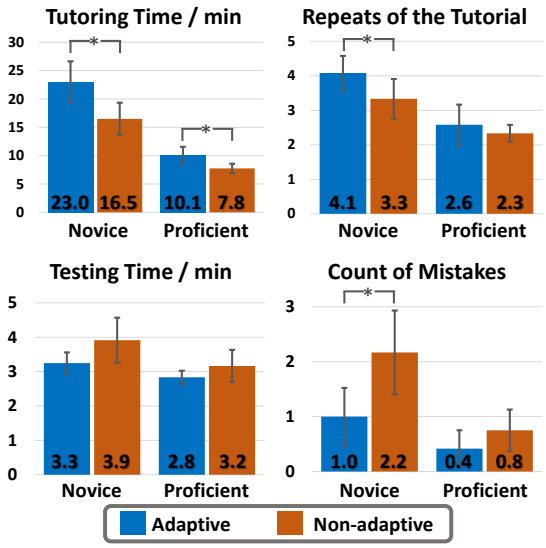


Figure 8: Objective performance during the tutoring and testing sections for novice and proficient participants. ($*$ = $p < .05$).

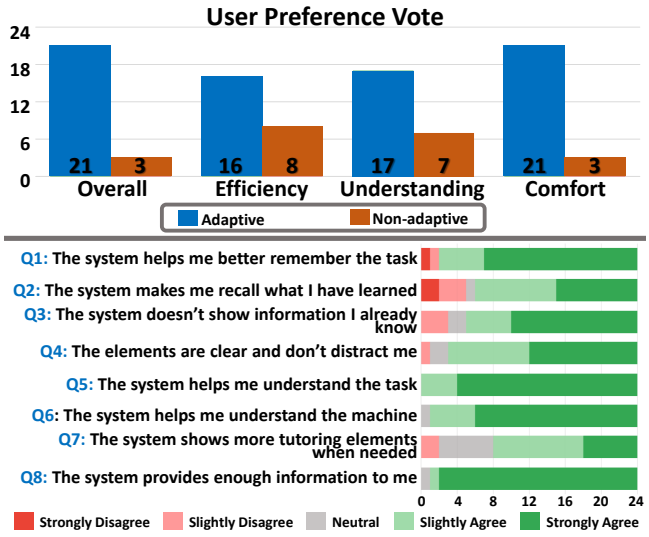


Figure 9: User votes and ratings for the features of the adaptive system.

what to do next." (P16) Meanwhile, the users acknowledged that the *adaptive* tutorial can properly hide the redundant information that was not needed and consequently the AR visualization is clear and non-distractive, especially compared with the *non-adaptive* tutorial. (Q3: $M = 4.2, SD = 1.0$, Q4: $M = 4.3, SD = 0.8$) "*Since I already went through the last session, (as a proficient user) I thought I didn't need that much tutoring element, and the adaptive system hided those not needed." (P7)*,"*With the adaptive system, my view was clearer because there were less AR elements distracting me." (P6)* The users also appreciated the adaptive feature which helped them to understand the task (Q5: $M = 4.8, SD = 0.4$, Q6: $M = 4.7, SD = 0.5$). "*After the avatar was hidden, I started to pay attention to the descriptions and got to understand the logic behind each machine operation." (P1)*,"*When the adaptive tutorial let me do it by myself, my brain was active and trying to understand the logic between the steps." (P15)* Moreover, it was receptive by the users that the *adaptive* tutorial made them better remember the tasks (Q1: $M = 4.5, SD = 1.0$, Q2: $M = 3.8, SD = 1.3$). "*The adaptive tutorial was gradually increasing the difficulty, which force me to remember the steps." (P14)*, "*Although I wasn't sure about the parameters of that step, I tried to perform it by myself and succeeded. This experience gave me a impression of that step." (P16)*

8 DISCUSSION, LIMITATIONS AND FUTURE WORK

In this section we discuss the primary results of the study and also provide design recommendations and insights for future adaptive tutoring systems.

Design of LoD. The design of LoD was first inspired by the formative study finding *(F2)*, and further proved to be receptive through the user study. The users agreed that the arrangement of the LoD fullfilled the needs at different learning stages. Yet, some users raised an interesting point. *"First, I pretty like the decrease of the tutoring elements as I learned more. But I wonder if the system could change the performance of the avatar or the animation according to my performance." (P14)* It reminds us that our adaptation model could also drive the modification of the tutoring elements in each LoD, not just hide or show. By leveraging the contextual visualization of AR, the adaptation model could foster more flexible designs of AR content for future adaptive tutoring systems.

Clear view of the adaptive system. The user study results illustrate that the adaptive system can provide the exactly necessary information according to the learning progress. *"I think in most of the time, the elements showed to me are what I needed. Only when I forgot something, it showed me more." (P2)* Additionally, some users addressed another advantage of adaptively displaying the tutoring elements. *"Compared to the first (non-adaptive) system, not showing the avatar and virtual animations really increased my learning speed. Because if they were always there, then I tended to dodge them, and they really distracted me." (P8)*

Adaptive system reduces over-confidence of novice users. Subjective ratings in confidence and memorization had no significant difference between adaptive and non-adaptive system, which meant they were all confident in remembering each step (Figure 7). However, objective performance in testing errors showed significant difference, especially for novice users (Figure 8). It revealed that novice users tended to be over-confident using non-adaptive system where all tutoring elements were always visible. *"I fully support the adaptive system because when I first used the non-adaptive one, I thought I remember everything. But I messed it up in testing. But for adaptive one, it forced me to remember and recall by hiding some elements." (P11)* Yet, proficient users clearly knew which steps needed more attention, so they had more accurate self-assessment. This expertise-dependent variation enlightens a potential research direction to developing a more sophisticated adaptation model to better assess a learner's performance.

Adaptation timing. In our system, when to display additional tutoring elements to the learners is determined by the time-sensitive adaptation model. Most users welcomed the feature where the system showed hints after they got stuck for a short period of time. However, two users mentioned that the tutoring elements sometimes appear too quickly while three users mentioned that the tutoring elements appear too late. In addition, two users mentioned that the timing of appearing tutoring elements should be more flexible. *"Maybe at the beginning, the hint can appear faster. And later, the hint appears slower for me to recall." (P18)* One potential solution is to add manual control for users to manage tutoring elements, such as using gestures (swiping their hand near the target component to uncover more details) or using voice command. Meanwhile, the system can collect the timing of manual control to fine-tune the thresholds of the adaptation model. For example, when hints are not shown, some users tend to recall without disruption, so the thresholds could be increased. In contrast, some users tend to see the hints more eagerly and may use manual control, so the thresholds could be decreased accordingly. By gradually collecting more data during the tutoring process, the system can minimize the need for users to do manual control.

The patterns of LoD change. During the user study, we logged the change of LoD for each user in the adaptive session. Referring to Figure 10 (left), we noticed it took three trials for the proficient users to reach level-2 LoD while four trials were taken for the novices to reach level-3. The results align with our expectation because the change of LoD is determined by the learners' real-time performance. The better a learner performs, the quicker the LoD decreases. Such pattern of LoD variation can be used to analyze the learners' performance and also fine-tune the adaptation model for further personalization.

The reasons of LoD increment. We counted the total occurrences of LoD increment (i.e., from i to i+1) for novice and proficient users in the adaptive session and grouped them by reasons. Referring to Figure 10 (right), we noticed that the novice users' LoDs were mostly incremented due to the unawareness of the task, while the proficient users' LoDs were mostly incremented due to interactions. This suggested that the proficient users tended to directly operate the target that they felt correct, while novice users tended to spend more time in observation. Such differences can be taken into account in developing the future adaptive systems.

Evaluating AR system in VR. Under COVID-19 situation, we conducted a remote VR study to evaluate AR features of the system, which was inspired by prior works [6, 36]. Admittedly, due to the difference between the accuracy of machine state and hand touching recognition in AR (89.1% and 93.4%) and VR (both 100%), the user study may miss some findings caused by the failure cases of recognition. However, the reported accuracy of low-level prediction in section 6.3 was only based on a single prediction, which was not the final accuracy to be leveraged in the high-level state prediction. We adopted a majority voting mechanism in which each prediction was decided by 5 consecutive predictions. The accuracy of machine state and hand touching recognition was increased (≈93% and ≈96%, respectively), and thus reduced the gap between the AR and VR evaluation. Moreover, since many proposed features (e.g., high-level recognition, LoD design) can be evaluated orthogonal to the low-level recognition, we can still obtain key findings from

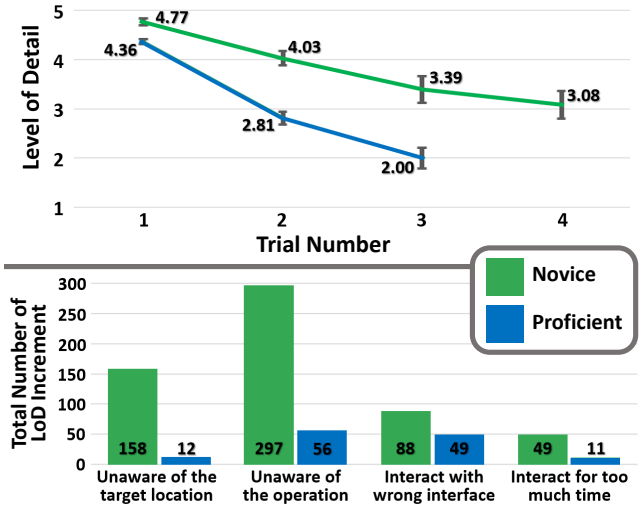


Figure 10: The patterns and reasons of LoD changes. (top) The average LoD of each trail for novice and proficient users in tutoring section. (bottom) The total number of LoD increment (i.e., from i to $i+1$) grouped by reasons for novice and proficient users.

the participants (e.g., the preference of adaptive vs. non-adaptive systems, patterns of LoD change, adaptation timing, etc). In retrospect, a better approach would be to add some random failure to the low-level recognition of VR system to simulate the AR system.

Generalizability of the system. Firstly, our system supports three common types of machine tasks: local, spatial, and body-coordinated interactions [27, 53]. Many manufacturing contexts are a combination of these three types of tasks (e.g., machine tools and CNC machines) [7]. Secondly, the recognition algorithm based on images can be applied to various machines. For example, nine common types of machine components (e.g., knobs, levers) were covered in the preliminary evaluation. Thirdly, the workflow design of chaining low-level and high-level recognition can be adapted to future systems of machine tasks, rather than our system alone.

Hardware and deep learning setup. Currently, our CNN model for machine state recognition works for *discrete* states of components. We envision that more robust image based object recognition networks, Internet of Things, and hand gesture and body skeleton detection systems in the near future can provide more accurate and plentiful input information to the adaptation model.

9 CONCLUSION

In this paper, we proposed an adaptation model that can automatically adjust the level of detail of AR tutoring elements. The model takes the input from the user and environment and performs low-level and high-level state prediction based on deep neural network and finite state machine. We also developed AdapTutAR, an AR-based adaptive tutoring system for *machine task*s that allows task authoring and tutoring via bodily demonstration. We evaluated the accuracy of the low-level state recognition on a mockup machine

with 9 component types, and further evaluated the overall adaptation model via a remote user study in VR environment. In the user study, we invited 24 participants to learn tutorials using adaptive and non-adaptive systems and collected their subjective ratings and objective performance. Based on the results, we believe that AdapTutAR provides important insights for future researchers in creating an adaptive tutoring system which empowers an efficient, flexible, and productive workforce.

ACKNOWLEDGMENTS

We wish to give a special thanks to the reviewers for their invaluable feedback. This work is partially supported by NSF under the grants FW-HTF 1839971, OIA 1937036, and CRI 1729486. We also acknowledge the Feddersen Chair Funds. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the funding agency.

REFERENCES

[1] 2020. Oculus. https://www.oculus.com/.

[2] Andrea F. Abate, Vincenzo Loia, Michele Nappi, Stefano Ricciardi, and Enrico Boccola. 2008. ASSYST: Avatar baSed SYStem mainTenance. *2008 IEEE Radar Conference, RADAR 2008* 1 (2008). https://doi.org/10.1109/RADAR.2008.4720962

[3] M. Beyyoudh, M. K. Idrissi, and S. Bennani. 2018. A new approach of designing an intelligent tutoring system based on adaptive workflows and pedagogical games. In *2018 17th International Conference on Information Technology Based Higher Education and Training (ITHET)*. 1–7. https://doi.org/10.1109/ITHET.2018.8424619

[4] Andrew Thomas Bimba, Norisma Idris, Ahmed Al-Hunaiyyan, Rohana Binti Mahmud, and Nor Liyana Bt Mohd Shuib. 2017. Adaptive feedback in computer-based learning environments: a review. *Adaptive Behavior* 25, 5 (2017), 217–234. https://doi.org/10.1177/1059712317727590

[5] Peter Brusilovsky and Hoah-Der Su. 2002. Adaptive Visualization Component of a Distributed Web-Based Adaptive Educational System. In *Intelligent Tutoring Systems*, Stefano A. Cerri, Guy Gouardères, and Fàbio Paraguaçu (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 229–238.

[6] Alisa Burova, John Mäkelä, Jaakko Hakulinen, Tuuli Keskinen, Hanna Heinonen, Sanni Siltanen, and Markku Turunen. 2020. Utilizing VR and Gaze Tracking to Develop AR Solutions for Industrial Maintenance. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.

[7] Yuanzhi Cao, Xun Qian, Tianyi Wang, Rachel Lee, Ke Huo, and Karthik Ramani. [n.d.]. An Exploratory Study of Augmented Reality Presence for Tutoring Machine Tasks. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA, 2020-04-21) *(CHI '20)*. Association for Computing Machinery, 1–13. https://doi.org/10.1145/3313831.3376688

[8] Jean-Rémy CHARDONNET, Guillaume Fromentin, and José OUTEIRO. 2017. Augmented reality as an aid for the use of machine tools. In *15th Management and Innovative Technologies (MIT) Conference*. Sinaia, Romania, 1–4. https://hal.archives-ouvertes.fr/hal-01598613

[9] Pei-Yu Chi, Sally Ahn, Amanda Ren, Mira Dontcheva, Wilmot Li, and Björn Hartmann. 2012. MixT: automatic generation of step-by-step mixed media tutorials. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. 93–102.

[10] Albert T. Corbett, Kenneth R. Koedinger, and John R. Anderson. 1997. Chapter 37 - Intelligent Tutoring Systems. In *Handbook of Human-Computer Interaction (Second Edition)* (second edition ed.), Marting G. Helander, Thomas K. Landauer, and Prasad V. Prabhu (Eds.). North-Holland, Amsterdam, 849 – 874. https://doi.org/10.1016/B978-044481862-1.50103-5

[11] F. De Crescenzio, M. Fantini, F. Persiani, L. Di Stefano, P. Azzari, and S. Salti. 2011. Augmented Reality for Aircraft Maintenance Training and Operations Support. *IEEE Computer Graphics and Applications* 31, 1 (2011), 96–101.

[12] Paula J. Durlach. 2019. Fundamentals, Flavors, and Foibles of Adaptive Instructional Systems. In *Adaptive Instructional Systems*, Robert A. Sottilare and Jessica Schwarz (Eds.). Springer International Publishing, Cham, 76–95.

[13] Pavel Dvorak, Radovan Josth, and Elisabetta Delponte. 2017. Object State Recognition for Automatic AR-Based Maintenance Guidance. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 1244–1250. https://doi.org/10.1109/CVPRW.2017.164 ISSN: 2160-7516.

[14] Andreas Fender, Philipp Herholz, Marc Alexa, and Jörg Müller. 2018. OptiSpace: Automated Placement of Interactive 3D Projection Mapping Content. 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, USA, Article 269, 11 pages. https://doi.org/10.1145/3173574.3173843

[15] Andreas Fender, David Lindlbauer, Philipp Herholz, Marc Alexa, and Jörg Müller. 2017. HeatSpace: Automatic Placement of Displays by Empirical Analysis of User Behavior. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 611–621. https://doi.org/10.1145/3126594.3126621

[16] Christoph Froschl. 2005. User modeling and user profiling in adaptive e-learning systems. *Graz, Austria: Master Thesis* (2005).

[17] Markus Funk. 2016. Augmented reality at the workplace : a context-aware assistive system using in-situ projection. http://dx.doi.org/10.18419/opus-8997

[18] Benjamin Goldberg, Keith Brawner, and Robert Sottilare. 2012. Use of Evidence-based Strategies to Enhance the Extensibility of Adaptive Tutoring Technologies. *Proceedings of the Interservice /Industry Training, Simulation, and Education Conference (I/ITSEC)* 12288 (2012), 1–12. http://ntsa.metapress.com/index/QHM8976477503684.pdf

[19] Dominic Gorecky, Mathias Schmitt, Matthias Loskyll, and Detlef Zühlke. 2014. Human-machine-interaction in the industry 4.0 era. In *2014 12th IEEE international conference on industrial informatics (INDIN)*. IEEE, 289–294.

[20] Michihiko Goto, Yuko Uematsu, Hideo Saito, Shuji Senda, and Akihiko Iketani. 2010. Task support system by displaying instructional video onto AR workspace. In *2010 IEEE International Symposium on Mixed and Augmented Reality*. IEEE, 83–90.

[21] Fernando Gutierrez and John Atkinson. 2011. Adaptive feedback selection for intelligent tutoring systems. *Expert Systems with Applications* 38, 5 (2011), 6146 – 6152. https://doi.org/10.1016/j.eswa.2010.11.058

[22] Ping-Hsuan Han, Yang-Sheng Chen, Yilun Zhong, Han-Lei Wang, and Yi-Ping Hung. 2017. My Tai-Chi Coaches: An Augmented-Learning Tool for Practicing Tai-Chi Chuan. In *Proceedings of the 8th Augmented Human International Conference* (Silicon Valley, California, USA) *(AH '17)*. Association for Computing Machinery, New York, NY, USA, Article 25, 4 pages. https://doi.org/10.1145/3041164.3041194

[23] Zhen-Yu He and Lian-Wen Jin. 2008. Activity recognition from acceleration data using AR model representation and SVM. In *2008 international conference on machine learning and cybernetics*, Vol. 4. IEEE, 2245–2250.

[24] S. J. Henderson and S. Feiner. 2009. Evaluating the benefits of augmented reality for task localization in maintenance of an armored personnel carrier turret. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. 135–144.

[25] S. J. Henderson and S. K. Feiner. 2011. Augmented reality in the psychomotor phase of a procedural task. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. 191–200. https://doi.org/10.1109/ISMAR.2011.6092386

[26] Bradley Herbert, Barrett Ens, Amali Weerasinghe, Mark Billinghurst, and Grant Wigley. 2018. Design considerations for combining augmented reality with intelligent tutors. *Computers and Graphics (Pergamon)* 77 (2018), 166–182. https://doi.org/10.1016/j.cag.2018.09.017

[27] Jean-Michel Hoc. 2001. Towards a Cognitive Approach to Human-Machine Cooperation in Dynamic Situations. *International Journal of Human-Computer Studies* 54, 4 (2001), 509–540. https://doi.org/10.1006/ijhc.2000.0454

[28] Lars-Erik Janlert. 2014. The Ubiquitous Button. *Interactions* 21, 3 (May 2014), 26–33. https://doi.org/10.1145/2592234

[29] Karen Kear, Frances Chetwynd, Judith Williams, and Helen Donelan. 2012. Web conferencing for synchronous online tutorials: Perspectives of tutors using a new medium. *Computers & Education* 58, 3 (2012), 953–963.

[30] Seungwon Kim, Gun Lee, Weidong Huang, Hayun Kim, Woontack Woo, and Mark Billinghurst. 2019. Evaluating the Combination of Visual Communication Cues for HMD-Based Mixed Reality Remote Collaboration. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300403

[31] A. Kotranza, D. Scott Lind, C. M. Pugh, and B. Lok. 2009. Real-time in-situ visual feedback of task performance in mixed environments for learning joint psychomotor-cognitive tasks. In *2009 8th IEEE International Symposium on Mixed and Augmented Reality*. 125–134. https://doi.org/10.1109/ISMAR.2009.5336485

[32] Wallace S. Lages and Doug A. Bowman. 2019. Walking with Adaptive Augmented Reality Workspaces: Design and Usage Patterns. In *Proceedings of the 24th International Conference on Intelligent User Interfaces* (Marina del Ray, California) *(IUI '19)*. Association for Computing Machinery, New York, NY, USA, 356–366. https://doi.org/10.1145/3301275.3302278

[33] Michael Laielli, James Smith, Giscard Biamby, Trevor Darrell, and Bjoern Hartmann. [n.d.]. LabelAR: A Spatial Guidance Interface for Fast Computer Vision Image Collection. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA, 2019-10-17) *(UIST '19)*. Association for Computing Machinery, 987–998. https://doi.org/10.1145/3332165.3347927

[34] Shaun W. Lawson and John R. G. Pretlove. 1998. Augmented reality for underground pipe inspection and maintenance. In *Telemanipulator and Telepresence Technologies V*, Matthew R. Stein (Ed.), Vol. 3524. International Society for Optics and Photonics, SPIE, 98 – 104. https://doi.org/10.1117/12.333673

[35] Jung Lee and O Park. 2008. Adaptive instructional systems. *Handbook of research on educational communications and technology* (2008), 469–484.

[36] David Lindlbauer, Anna Maria Feit, and Otmar Hilliges. 2019. Context-Aware Online Adaptation of Mixed Reality Interfaces. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 147–160. https://doi.org/10.1145/3332165.3347945

[37] Matthias Loskyll, Ines Heck, Jochen Schlick, and Michael Schwarz. 2012. Context-based orchestration for control of resource-efficient manufacturing processes. *Future Internet* 4, 3 (2012), 737–761.

[38] Danielle L. Lusk, Amber D. Evans, Thomas R. Jeffrey, Keith R. Palmer, Chris S. Wikstrom, and Peter E. Doolittle. 2009. Multimedia learning and individual differences: Mediating the effects of working memory capacity with segmentation. *British Journal of Educational Technology* 40, 4 (2009), 636–651. https://doi.org/10.1111/j.1467-8535.2008.00848.x

[39] Stephen Maloney, Michael Storr, Sophie Paynter, Prue Morgan, and Dragan Ilic. 2013. Investigating the efficacy of practical skill teaching: a pilot-study comparing three educational methods. *Advances in Health Sciences Education* 18, 1 (2013), 71–80.

[40] Paula Martiskainen, Mikko Järvinen, Jukka-Pekka Skön, Jarkko Tiirikainen, Mikko Kolehmainen, and Jaakko Mononen. 2009. Cow behaviour pattern recognition using a three-dimensional accelerometer and support vector machines. *Applied animal behaviour science* 119, 1-2 (2009), 32–38.

[41] Michael A Orey and Wayne A Nelson. 1993. Development principles for intelligent tutoring systems: Integrating cognitive theory into the development of computer-based instruction. *Educational Technology Research and Development* 41, 1 (1993), 59–72. https://doi.org/10.1007/BF02297092

[42] Philo Tan Chua, R. Crivella, B. Daly, Ning Hu, R. Schaaf, D. Ventura, T. Camill, J. Hodgins, and R. Pausch. 2003. Training for physical tasks in virtual environments: Tai Chi. In *IEEE Virtual Reality, 2003. Proceedings.* 87–94.

[43] Thammathip Piumsomboon, Gun A. Lee, Jonathon D. Hart, Barrett Ens, Robert W. Lindeman, Bruce H. Thomas, and Mark Billinghurst. 2018. Mini-Me: An Adaptive Avatar for Mixed Reality Remote Collaboration. 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, USA, Article 46, 13 pages. https://doi.org/10.1145/3173574.3173620

[44] Thammathip Piumsomboon, Gun A. Lee, Andrew Irlitti, Barrett Ens, Bruce H. Thomas, and Mark Billinghurst. 2019. On the Shoulder of the Giant: A Multi-Scale Mixed Reality Collaboration with 360 Video Sharing and Tangible Interaction. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, Article 228, 17 pages. https://doi.org/10.1145/3290605.3300458

[45] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. [n.d.]. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *Advances in Neural Information Processing Systems 28*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett (Eds.). Curran Associates, Inc., 91–99. http://papers.nips.cc/paper/5638-faster-r-cnn-towards-real-time-object-detection-with-region-proposal-networks.pdf

[46] Alejandro Reyes, Osslan Vergara, Erasmo Bojórquez, Vianey Sánchez, and Manuel Nandayapa. 2016. A mobile augmented reality system to support machinery operations in scholar environments: A MAR SYSTEM TO SUPPORT MACHINERY OPERATIONS IN SCHOLAR ENVIRONMENTS. *Computer Applications in Engineering Education* (10 2016). https://doi.org/10.1002/cae.21772

[47] D. Rodenburg, P. Hungler, S. A. Etemad, D. Howes, A. Szulewski, and J. Mclellan. 2018. Dynamically adaptive simulation based on expertise and cognitive load. In *2018 IEEE Games, Entertainment, Media Conference (GEM)*. 1–6.

[48] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[49] Eldon Schoop, Michelle Nguyen, Daniel Lim, Valkyrie Savage, Sean Follmer, and Björn Hartmann. 2016. Drill Sergeant: Supporting physical construction projects through an ecosystem of augmented tools. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. 1607–1614.

[50] Marieke H.S.B. Smits, Jo Boon, Dominique M.A. Sluijsmans, and Tamara van Gog. 2008. Content and timing of feedback in a web-based learning environment: effects on learning as a function of prior knowledge. *Interactive Learning Environments* 16, 2 (2008), 183–193. https://doi.org/10.1080/10494820701365952

[51] Robert Sottilare and Keith Brawner. [n.d.]. Component Interaction within the Generalized Intelligent Framework for Tutoring (GIFT) as a Model for Adaptive Instructional System Standards Toward Standardization through Design Goals. ([n. d.]), 55–62.

[52] Robert A. Sottilare, Benjamin Goldberg, Charles Ragusa, and Michael Hoffman. 2013. Characterizing an Adaptive Tutoring Learning Effect Chain for Individual and Team Tutoring. 13033 (2013), 1–13.

[53] Lucy Suchman. 2007. *Human Machine Reconfigurations Plans and Situated Actions* (2 ed.). Cambridge University Press.

[54] Alyssa Tanaka, Jeffrey Craighead, Glenn Taylor, and Robert Sottilare. 2019. Adaptive Learning Technology for AR Training: Possibilities and Challenges. In *Adaptive Instructional Systems*, Robert A. Sottilare and Jessica Schwarz (Eds.). Springer International Publishing, Cham, 142–150.

[55] Markus Tatzgern, Valeria Orso, Denis Kalkofen, Giulio Jacucci, Luciano Gamberini, and Dieter Schmalstieg. 2016. Adaptive information density for augmented reality displays. *Proceedings - IEEE Virtual Reality* 2016-July, 83 – 92. http://dx.doi.org/10.1109/VR.2016.7504691 Adaptive information;Browser interfaces;Hierarchical clustering;Level of detail;Search tasks;Semantic attribute;User interaction;Visual clutter;.

[56] Andrea L. Thomaz and Cynthia Breazeal. 2008. Teachable robots: Understanding human teaching behavior to build more effective robot learners. *Artificial Intelligence* 172, 6-7 (2008), 716–737. https://doi.org/10.1016/j.artint.2007.09.009

[57] Balasaravanan Thoravi Kumaravel, Fraser Anderson, George Fitzmaurice, Bjoern Hartmann, and Tovi Grossman. 2019. Loki: Facilitating Remote Instruction of Physical Tasks Using Bi-Directional Mixed-Reality Telepresence. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (New Orleans, LA, USA) *(UIST '19)*. Association for Computing Machinery, New York, NY, USA, 161–174. https://doi.org/10.1145/3332165.3347872

[58] Balasaravanan Thoravi Kumaravel, Cuong Nguyen, Stephen DiVerdi, and Björn Hartmann. 2019. TutoriVR: A Video-Based Tutorial System for Design Applications in Virtual Reality. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. 1–12.

[59] Unity. 2019. Unity Real-Time Development Platform. Retrieved September 1, 2019 from https://unity.com/.

[60] Kurt VanLehn. 2011. The relative effectiveness of human tutoring, intelligent tutoring systems, and other tutoring systems. *Educational Psychologist* 46, 4 (2011), 197–221. https://doi.org/10.1080/00461520.2011.611369

[61] Anne Wegerich and Matthias Rötting. 2011. A Context-Aware Adaptation System for Spatial Augmented Reality. In *Digital Human Modeling*, Vincent G. Duffy (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 417–425.

[62] Giles Westerfield, Antonija Mitrovic, and Mark Billinghurst. 2015. Intelligent Augmented Reality Training for Motherboard Assembly. *International Journal of Artificial Intelligence in Education* 25, 1 (2015), 157–172. https://doi.org/10.1007/s40593-014-0032-x

[63] Frederik Winther, Linoj Ravindran, Kasper Paabøl Svendsen, and Tiare Feuchtner. 2020. Design and Evaluation of a VR Training Simulation for Pump Maintenance Based on a Use Case at Grundfos. In *IEEE VR 2020*. IEEE.

[64] Li Da Xu, Eric L Xu, and Ling Li. 2018. Industry 4.0: state of the art and future trends. *International Journal of Production Research* 56, 8 (2018), 2941–2962.

[65] M. Yamashita and S. Sakane. 2001. Adaptive annotation using a human-robot interface system PARTNER. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, Vol. 3. 2661–2667 vol.3.

[66] S. Yonemoto. 2013. Seamless Annotation Display for Augmented Reality. In *2013 International Conference on Cyberworlds*. 387–387.

[67] J. Zhu, S.K. Ong, and A.Y.C. Nee. 2015. A context-aware augmented reality assisted maintenance system. *International Journal of Computer Integrated Manufacturing* 28, 2 (2015), 213–225. https://doi.org/10.1080/0951192X.2013.874589

[68] Z. Zhu, V. Branzoi, M. Wolverton, G. Murray, N. Vitovitch, L. Yarnall, G. Acharya, S. Samarasekera, and R. Kumar. 2014. AR-mentor: Augmented reality based mentoring system. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 17–22.