

SIGMACOLLAB: AN APPLICATION-DRIVEN DATASET FOR PHYSICALLY SITUATED COLLABORATION

Dan Bohus¹, Sean Andrist¹, Ann Paradiso¹, Nick Saw¹, Tim Schoonbeek^{*2}, and Maia Stiber¹

¹Microsoft Research, Redmond, WA, USA

²Eindhoven University of Technology, Eindhoven, Netherlands

{dbohus; sandrist; annparadiso; nicksaw}@microsoft.com; t.schoonbeek@tue.nl; maiastiber@microsoft.com

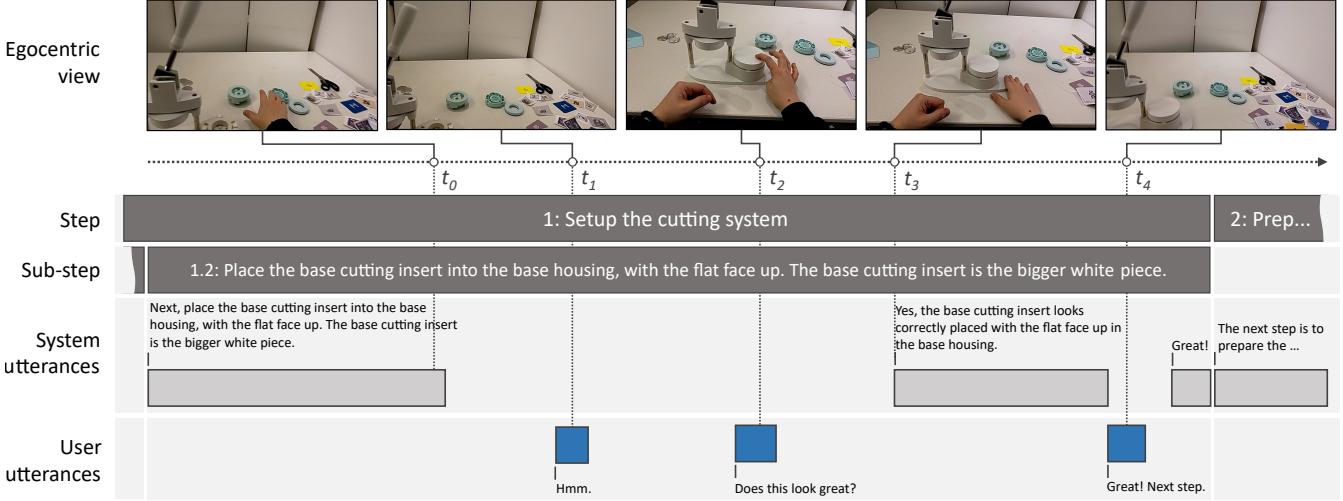


Figure 1: Schematic illustration of data streams from the SIGMACOLLAB dataset in which the SIGMA AI agent assists a participant to create a pin-back button using a button press machine. The transcript of the full interaction is shown in Appendix A1.

ABSTRACT

We introduce SIGMACOLLAB, a dataset enabling research on physically situated human-AI collaboration. The dataset consists of a set of 85 sessions in which untrained participants were guided by a mixed-reality assistive AI agent in performing procedural tasks in the physical world. SIGMACOLLAB includes a set of rich, multimodal data streams, such as the participant and system audio, egocentric camera views from the head-mounted device, depth maps, head, hand and gaze tracking information, as well as additional annotations performed post-hoc. While the dataset is relatively small in size (≈ 14 hours), its application-driven and interactive nature brings to the fore novel research challenges for human-AI collaboration, and provides more realistic testing grounds for various AI models operating in this space. In future work, we plan to use the dataset to construct a set of benchmarks for physically situated collaboration in mixed-reality task assistive scenarios. SIGMACOLLAB is available at <https://github.com/microsoft/SigmaCollab>.

1 INTRODUCTION

Fluid human-machine interaction has been a decades-long pursuit in the research community. Building computational systems that can engage in interaction and collaborate with people *in the physical world*, such as virtual assistive agents, interactive robots, or mixed-reality assistants brings forward multiple challenges at the intersection of several research fields: artificial intelligence, computer vision, natural language processing, and human-computer interaction.

Physically situated systems need to make sense (in real time and to the extent needed by each specific application) of the surrounding physical environment and of the actions performed by humans in the environment. This often involves a set of com-

petencies such as self-localization and mapping, object detection and tracking, depth estimation, face detection, body pose tracking, action recognition, and so forth. Many of these tasks have been extensively studied in the computer vision research community, with significant progress over the last two decades. The recent emergence of foundation models and large multimodal models has enabled a transition towards open-domain computer vision, and, by tying the models to language, has enabled reasoning about and answering more complex questions about the physical world. A lot of this progress has been made possible by the collection and curation of various datasets and data resources that have been used in the community to train and to comparatively evaluate the performance of various algorithms.

*Work conducted as part of an internship at Microsoft Research

Situated collaboration raises, however, additional *interaction-related* challenges. In addition to understanding the environment, objects, and actions, creating seamless situated collaborations requires fine-grained temporal coordination that is grounded in a deep understanding of human cognitive states such as, intentions, goals, beliefs, and various affective states. In these areas, progress has been slower. Today, most systems still use a "ping-pong" model of interaction where one user question is followed in a linear manner by one system answer, which limits naturalness and curtails opportunities for fluid, mixed-initiative collaboration. Part of the difficulty with interaction-related challenges lies in reifying, formalizing, and measuring the cognitive states and interactive phenomena involved. Moreover, the traditional approach of using collected, static datasets to build and evaluate models works only to a certain degree when it comes to interaction. Any trained model, once deployed, may alter user behavior, shifting the distribution of the underlying data. The study and evaluation of interactive phenomena are best done in the context of actual interactive systems.

To foster and enable more research on interaction and collaboration challenges, we introduce a new resource: an *application-driven, interactive dataset* called SIGMACOLLAB. The dataset consists of 85 interactive sessions in which untrained participants interacted with a mixed-reality assistive AI application which guided them in performing certain tasks in the physical world, such as binding a notebook or installing the wheels on a skateboard. The dataset is egocentric and multimodal in nature; it includes audio, egocentric camera views, head, hands and gaze tracking, and it spans approximately 14 hours of interaction data.

SIGMACOLLAB's value-proposition lies in its *application-driven* and *interactive* nature. The data collection approach, in which participants interact with an AI application surfaces novel challenges that are not present in non-interactive datasets. As users engage with the system while pursuing meaningful and varied goals, the approach yields data with greater ecological validity. Finally, the open-source nature of the AI application used for collecting the data enables researchers to deploy models developed or evaluated based on this data in the context of the target application, and allows them to collect additional data to study their end-effects on metrics of overall task performance. We publish SIGMACOLLAB to catalyze rigorous, application-grounded study of fluid human–AI collaboration and to close the gap between lab benchmarks and real-world performance.

2 RELATED WORK

Publicly released academic datasets have historically played a transformative role in advancing computer vision research. The release of ImageNet [17] was a turning point for large-scale visual classification, enabling the development of powerful deep convolutional architectures. Subsequently, COCO [27] significantly pushed the field of object detection and segmentation, while Kinetics [12] has become a cornerstone for video understanding and action recognition, and Cityscapes [13] for pixel-level scene understanding. Beyond these general-purpose datasets, application-driven benchmarks have similarly accelerated progress in specialized domains. For example, MNIST [25] catalyzed early breakthroughs in digit recognition and thereby modern deep learning, KITTI [19] has been instrumental in autonomous driving research, and LFW [23] in face recogni-

tion. These examples highlight that introducing a well-designed, meaningful dataset in a specific application area can catalyze rapid advances in methodology, benchmarks, and downstream impact that can shape the direction of entire research fields.

Spurred by an increased interest in embodied interaction, mixed-reality and robotics, multiple datasets and benchmarks have been developed in recent years for egocentric computer vision settings. For example, the OpenEQA dataset [28] provides embodied question-answer pairs that are anchored in videos (or 3D-scans) of real world environments. The EPIC-KITCHENS project [14–16] captured daily activities in the kitchen from a first-person perspective, in support of challenges like action detection, action recognition, object segmentation, *etc.* The Ego4D dataset [20] includes over 3000 hours spanning hundreds of different indoors and outdoors activities, and defines various benchmarks for memory, forecasting and social understanding challenges. Later, the Ego-Exo4D [21] effort extends this by simultaneously capturing both ego and exo-centric views of skilled human activities and includes benchmarks for egocentric tracking of body and hand poses and for proficiency estimation. Other datasets like GTEA Gaze, GTEA Gaze+, EGTEA Gaze+ [18, 26] focused on capturing eye gaze in conjunction with egocentric views, and leveraging that signal on various action-related challenges. A number of other egocentric datasets have also been constructed around structured, procedural task execution or performance (rather than spontaneous activities), such as Assembly101 [33], CaptainCook4D [30], IndustReal [32], MECCANO [4], *etc.* These datasets support challenges like mistake detection, procedure step recognition, active object detection, action anticipation.

While these egocentric datasets provide a rich empirical testbed for many computer vision challenges, they generally capture a single actor performing an activity in the world, and are therefore not supportive of interaction or collaboration-related challenges. In contrast, two other recent datasets, HoloAssist [38] and WTaG [5] were collected in an interactive setting: in both cases a human instructor guided a human worker who was wearing a HoloLens mixed-reality headset through performing a physical task. In addition to typical computer vision challenges, these data also surface a variety of interesting problems around interaction-related challenges, such as proactive interventions, grounding, reference generation and resolution, *etc.* While these interactive datasets are certainly valuable, both involve interactions between two humans (an executor and an instructor). In contrast, SIGMACOLLAB is based on a stand-alone system that does not require human supervision. Consequently, its contents more accurately reflect challenges that arise in a realistic application.

3 AN APPLICATION-DRIVEN APPROACH

The datasets described above were collected using a diverse set of methods: scraping YouTube videos *e.g.* [11, 29, 39], generating data using 3D simulators *e.g.* [22, 28], capturing human activities with varying degrees of spontaneity or structure in lab *e.g.* [4, 18, 26, 32, 34] or in natural settings *e.g.* [14–16, 20, 21], capturing human-human interactions *e.g.* [5, 38], *etc.* The annotations that often supplement the data and provide the basis for evaluation typically involve labor-intensive human effort, crowd-sourcing, and more recently the use of large language

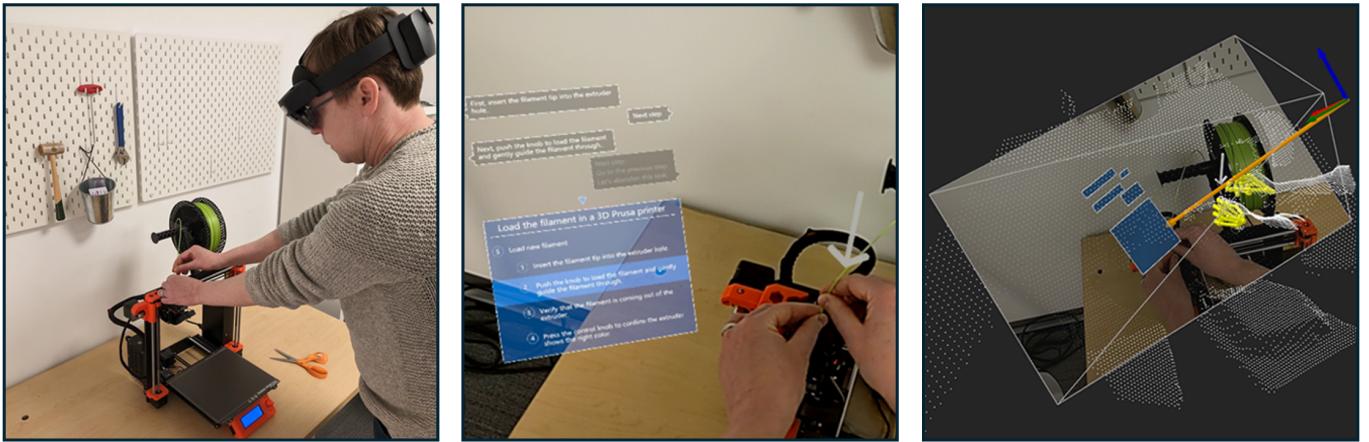


Figure 2: **Left:** user performing a procedural task with a mixed-reality headset running SIGMA. **Middle:** first-person view showing SIGMA guidance panel and task-specific holograms. **Right:** visualization of system’s scene understanding showing the egocentric camera view, depth map, detected objects, gaze, hand and head pose in 3D space. ©2024 IEEE

models. Each data collection and annotation approach colors the nature of the data, and problems that can be approached using it.

Our long term goal is to enable fluid human-computer interaction in physically situated settings. With this target in mind, we focus on the less-well addressed interaction-related challenges and we introduce an *interactive, application-driven approach* to dataset development. Specifically, we collect data by having participants interact with an existing AI application. The participants are motivated by accomplishing an end goal, and the interactions, *i.e.*, the data collected, are therefore situated in space and time within the context of a specific task.

This approach has a number of advantages. First, it leads to collecting data that is ecologically more valid, in the sense that the challenges reflected in the data are better matched to the end goal of supporting fluid collaboration. For example, in an LLM-based approach to embodied question-answer generation, the constructed questions are most likely fully-formed sentences, such as “*What is to the left of the microwave?*”, whereas the questions that naturally arise during an interaction may be fragments that contain a significantly larger number of referential expressions, such as “*Like this?*” or “*Is it this one?*”. At the same time, in contrast to data collected in human-human settings, the temporal patterns and content of utterances reflects the style people adopt when interacting with an AI agent, rather than with another human.

Second, the approach can bring to the fore new questions and challenges. For example, one observation in the SIGMACOLLAB dataset collected with this approach is that participants often talk to themselves throughout the task execution. This in turn raises the challenge of detecting self-talk, *i.e.*, understanding which utterances should be responded to and which not. This is a interaction-related competency that an agent should have, but that to our knowledge has not been addressed in the community.

Finally, because the framework relies on an open-source application, models developed or evaluated on this data can be integrated back into the original application, and tested in new live interactions to understand the full, end-to-end effects on task-level performance and user satisfaction metrics. This in

turn can surface new questions and challenges, within successive cycles of refinement.

In this work we use this interactive, application-driven approach to construct a dataset for physically situated collaboration. We use SIGMA [9, 10] as the target application — an open-source [3], mixed-reality task assistive research testbed system that guides the participants through the execution of various procedural tasks in the physical world. In the next section we describe SIGMA in more detail, including the system architecture and the data capture configuration. Next, in Section 5, we describe the data collection experiment and present basic dataset statistics. In Section 6 we discuss additional annotations and post-hoc processing performed on the data. Finally, in Section 7 we present conclusions and outline future work.

4 SIGMA

We begin with a brief description of SIGMA’s functionality and system architecture (an in-depth description and additional technical details are available in [9]). Next, we outline a number of adaptations made to the system in subsection 4.2 and discuss the specific configuration used during the data collection in subsection 4.3.

4.1 Overview

SIGMA is an open-source [3] system that was developed to serve as a testbed for research in mixed-reality task assistance [9, 10]. The system leverages a HoloLens 2 headset and can guide users step-by-step through performing procedural tasks in the physical world (see Figure 2).

Task recipes are organized linearly, into a series of steps with sub-steps, and can be generated automatically via an LLM or can be manually authored in a simple json format. Throughout the interactions, the system displays a virtual panel containing the task instructions, as shown in Figure 2, and reads aloud the instructions to the user one sub-step at a time. A history of the recent dialog is visible to the user in a floating virtual panel above the task recipe (see also Figure 2). In addition, SIGMA

may also render step-specific holograms, such as the downward pointing arrow shown in middle image from Figure 2. Once the user completes a sub-step, they can navigate to the next sub-step with a simple voice command like “*Next step*”. Users can also ask questions throughout the interaction, and other navigational commands allow going directly to a particular step and sub-step number or going backwards, to a previous step.

Figure 1 illustrates a snippet from a larger interaction in which a user is using a button press machine to create a pin-back button (the entire exchange is presented in Appendix A1). Once the system provides a sub-step instruction to the user, after each user utterance SIGMA first uses keyword spotting over the speech recognition results to identify simple user requests like “*next*” or “*previous step*”. If the keyword spotting process fails, the system relies on an LLM to decide how to respond to the user. Specifically, a *response-generating prompt* that leverages the dialog context, as well as visual information (the egocentric image at the beginning of the user utterance) is used to instruct an LLM to classify the last user utterance into one of four classes: **question** (the user is asking a question), **no-question** (self-talk, *i.e.*, the user is talking or making a comment to themselves), **next-step** (the user is making a statement indicating that they are ready to move on to the next step), or **step-navigation** (the user is issuing a more complex navigation command); additionally the prompt also instructs the LLM to provide an answer if the utterance is classified as a question. An example of the an instantiated prompt and corresponding response is shown in Appendix A2.

Architecturally, SIGMA is implemented as a distributed system: a lightweight client application runs on the HoloLens 2 device configured in Research Mode [37], and communicates live with a server application that runs on a desktop computer. The client application captures data from the various HoloLens sensors (*e.g.*, color, depth in long-throw mode, grayscale cameras, microphone) and streams them live to the server, which runs the bulk of the computation, including speech recognition, interaction management, speech synthesis, *etc*. The server application streams back to the client in real time instructions for actuating the user interface, and rendering holograms and audio. The on-device sensor streams, together with additional streams of data generated by the server, are persisted on the server side. For the interested reader, additional technical details of the system’s architecture and implementation are available in [9].

4.2 System Updates and Changes

Prior to data collection, we performed a number of pilot experiments with SIGMA and made a several changes and adjustments to the system based on lessons learned. Some, but not all, of these experiments are described in [6].

The original version of SIGMA, described in [9], used a text-only response-generating prompt. We updated the system to leverage multimodal models (such as the GPT-4o family of models) and use both text and images when responding to the user. We designed and tuned a more complex response-generating prompt that includes information about the task recipe, the step and sub-step the user is currently on, the history of the dialog in the current sub-step, as well as an image collected at the beginning of the user utterance from the right-side egocentric grayscale camera on the device. We chose one of the egocentric grayscale cameras for this because its field of view better captures the user’s hands than the color camera in the HoloLens2 device, which is pointed more upward and often misses important portions of the workspace. More details about the design of this prompt, together with an example are shown in Appendix A2.

To improve response latencies, we transitioned from the Azure Speech [1] service used by default in SIGMA and implemented a solution that combines the Silero voice activity detector [36] with the Whisper speech recognition model [31]. For Whisper, we selected the smallest and fastest variant, `tiny.en`, to optimize latency. We also implemented a continuation detector: it detects when a user begins speaking shortly after pausing, and merges the speech fragments into a single utterance for processing. In addition, we implemented support for canceling LLM requests that are in flight if the user starts speaking before the results from the LLM arrive. In this case, LLM query triggered by the first user utterance or fragment is canceled and a new query is issued.

Finally, we made several other performance-focused changes to optimize the streaming of sensor data from client to desktop. Other minor changes included improvements to support for step navigation, and minor user interface updates.

4.3 Data Capture Configuration

We configured the SIGMA client app to collect image views from the RGB, depth and the two front-facing grayscale cameras, at the resolutions and target frame-rates described in Table 1. For

Stream	Representation	Target Frame rate	Actual Avg. Frame rate
Color Camera View	896 × 504 pixels @ 24bpp, with camera pose and intrinsics	15 Hz	14.91 Hz
Depth Camera View (long-throw)	320 × 288 pixels @ 16bpp, with camera pose and intrinsics	5 Hz	4.98 Hz
Left Front Grayscale Camera View	640 × 480 pixels @ 8bpp, with camera pose and intrinsics	15 Hz	13.64 Hz
Right Front Grayscale Camera View	640 × 480 pixels @ 8bpp, with camera pose and intrinsics	15 Hz	13.64 Hz
Head Pose + Eye Gaze	tuple of head pose matrix (4 × 4) and eye gaze ray (3 × 1 origin position vector and 3 × 1 direction vector)	30 Hz	28.37 Hz
Hands Pose	pose matrices (4 × 4) for each of the 26 joints in the left and right hand	20 Hz	20.01 Hz
Audio	1-channel, 32-bit floating-point PCM	16 kHz	16.00 kHz

Table 1: Configuration for SIGMA streams: representation, target configured frame rates, as well as actual frame-rates during data collection.

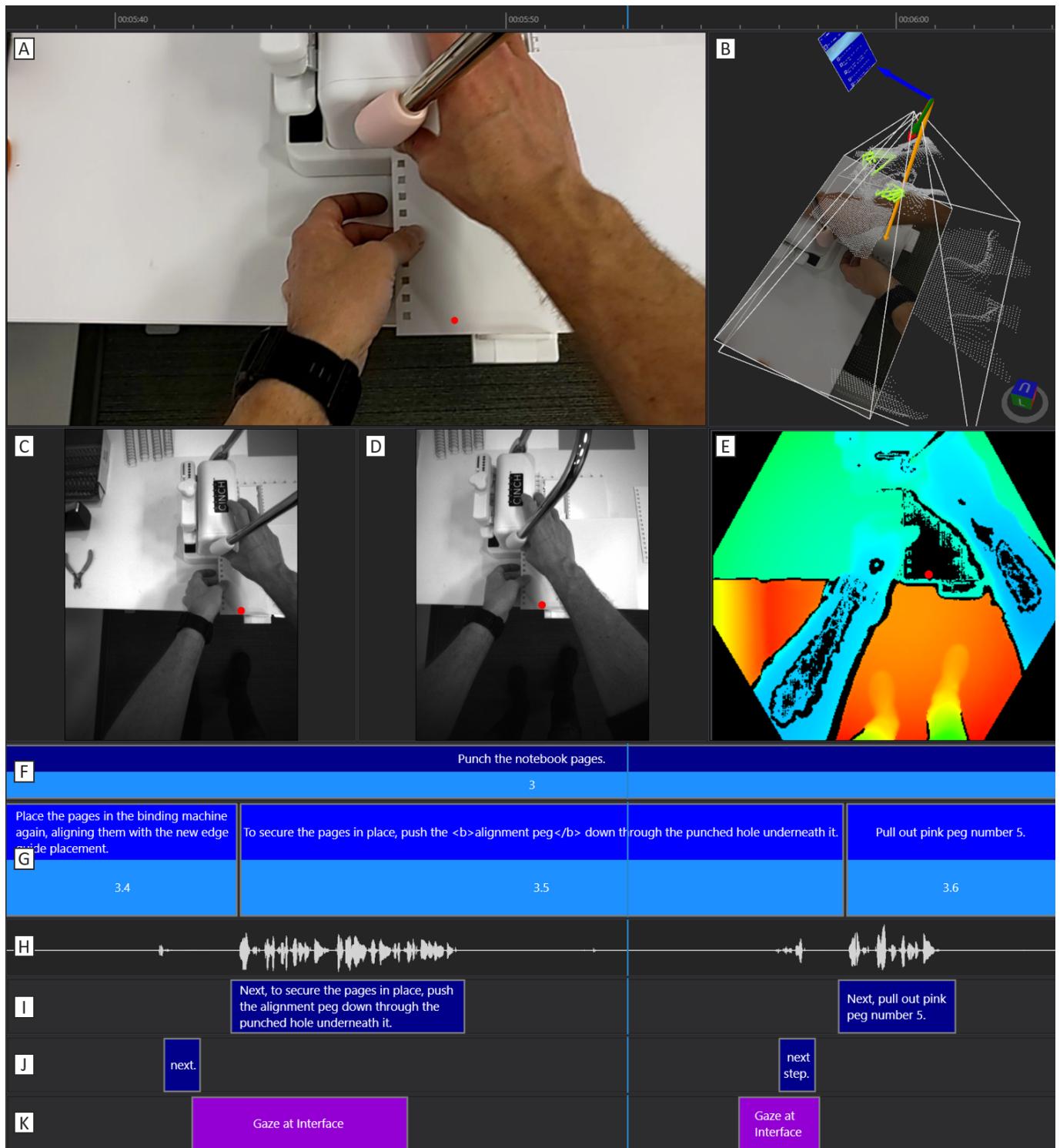


Figure 3: Illustration of the various data streams available in the SIGMACOLLAB dataset. Top of the image has the time axis and the blue vertical line represents the current time position for the image streams. (A) Color camera image; (B) 3D color and depth camera poses, head pose, gaze direction, hands, and mixed reality UI positioning; (C) left grayscale camera image; (D) right grayscale camera image; (E) depth image; (F) temporal step execution boundaries; (G) temporal sub-step execution boundaries; (H) audio; (I) system utterances; (J) participant utterances; (K) temporal intervals in which the participant is gazing at the mixed reality user interface panel. The red dots in (A), (C), (D), and (E) show the corresponding projected gaze points.

Task	Description	# Steps	# Sub-steps	Duration (mm:ss)
Nespresso	Make coffee using a Nespresso Pixie machine	4	10	03:33
Hard-drive	Replace a hard-drive in a PC	4	12	03:50
Skateboard	Install the wheels on a skateboard	5	23	10:26
Button	Make a button using a button press machine	6	27	08:40
Notebook	Make a notebook using a binding machine	5	34	10:07
Margarita	Make a Margarita mocktail	3	14	06:28
Mojito	Make a Mojito mocktail	4	17	07:39
Whiskey	Make a Whiskey Sour mocktail	5	18	07:44

Table 2: Tasks, together with number of steps and sub-steps and duration in the expert demonstration.

each camera, we also captured the camera intrinsics parameters, as well as the live camera position in 3D-space (relative to the same world anchor created in the lab used for data collection). Additionally, we captured audio from the device microphone array, as well as head, hand and gaze tracking information. The sensor data streams described in Table 1 provide a rich multimodal basis for constructing future challenges for situated collaboration. Figure 3 shows a sample visualization of the various streams available in the data.

Three different Azure GPT model deployments were used in conjunction with the response-generating prompt throughout the data collection study. In the first part of the study we relied on a GPT-4o model using a global data zone region². After observing relatively large response latencies, we switched to a GPT-4o-mini deployment in a US data zone region. Finally, for the last third of the dataset, we switched back to a GPT-4o deployment using a US data zone region. The resulting three sections of the dataset are described in Appendix A3.

5 SIGMACOLLAB DATASET

To assemble SIGMACOLLAB, we conducted a data collection experiment in which participants interacted with SIGMA to perform a set of procedural tasks. The experiment was approved by the Microsoft Research Institutional Review Board. Below, we describe the battery of tasks used, the data collection protocol and relevant statistics from the resulting dataset.

5.1 Tasks

We defined eight procedural tasks (see Table 2) to be used in the data collection study. The tasks span multiple domains. They include a simple task with a short number of sub-steps: making coffee using a Nespresso machine; two hardware tasks: installing wheels on a skateboard and changing a hard-drive in a computer; two craft tasks: making a pin-back button and making a notebook; and three mocktail preparation tasks: making a Margarita, making a Mojito, and making a Whiskey Sour.

We designed the task recipes in an iterative process, and brought further refinements based on observations made in pilot studies. We did not include any step-specific holograms in the task definitions, hence the only holograms visible to the user throughout the task execution are the task panel and dialog history. The final

set of recipes, together with some illustrative image captures for each task, are shown in Appendix A4.

The complexity of the tasks varies, as reflected by the number of steps and sub-steps in each recipe (see Table 2): the simplest task has a total of 10 sub-steps, and the longest has 34. When executed by an expert that simply performs each of the sub-steps correctly, at a regular pace, and without asking any questions, the durations vary from 3.5 minutes to over 10 minutes.

The tasks span a diverse range of objects, ingredients, materials, and tools. These items vary along several dimensions:

- Size: from very small (*e.g.*, screws, buttons) to medium-sized (*e.g.*, skateboard, Nespresso machine);
- Familiarity: some are common, everyday items (*e.g.*, screwdriver, lemon, coffee cup), while others are specialized tools unfamiliar to most people (*e.g.*, skate tool, cinching machine, Hawthorne strainer);
- Quantity: some are single items (*e.g.*, Nespresso machine, Boston shaker), others appear in multiples (*e.g.*, ice cube, Nespresso capsule, mint leaf), and some are treated as sets (*e.g.*, ice cubes, mint leaves);
- Material properties: the tasks involve both non-deformable and deformable objects, including various liquids.

Furthermore, not all objects present in the workspace are relevant to the task at hand. Overall, these dimensions of variability are similar to what one might encounter in real-world interactive scenarios, and increase the computer vision challenges present in this dataset.

The steps in the selected procedural tasks also exhibit a range of characteristics involving various kinds of physical actions to manipulate the relevant task objects. There are several different types of steps present across all eight tasks (the full mapping of all task steps to their specific category can be found in Appendix A4):

- Discrete Event: characterized by a crisp, instantaneous event that marks the successful completion of the step, *e.g.*, “Plug the wide connector into the hard-drive.”
- Move: an object is moved, removed, or placed in some position, possibly involving fuzzy micro-adjustments after the placement, *e.g.*, “Take a front and back cover

²geographical region for the datacenter providing the service

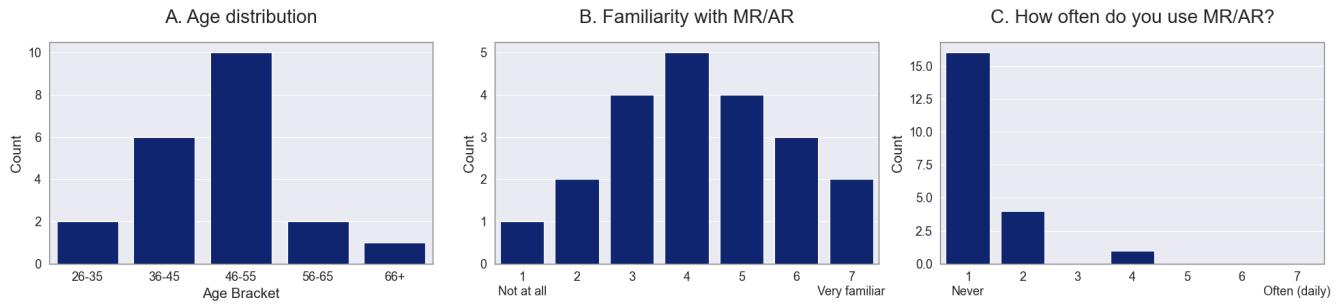


Figure 4: Data collection study participant demographics.

and place them in the binding machine, aligning with the edge guide.” This step might require small adjustments to accomplish the final alignment after placing.

- Add Amount: an incremental or continuous pouring, squeezing, or addition of a substance into a container, usually a liquid, *e.g.*, “Pour in 0.5 ounces of simple syrup.”
- Check: a conditional adjustment that may or may not involve action, *e.g.*, “Adjust the kingpin nut if necessary to achieve the desired tightness or looseness for turning.”
- Invisible State: a step where the final outcome is not directly observable, and is only inferable by the users’s completion of their action, *e.g.*, “Screw the case screw back using the screwdriver.” It is not directly observable from visible object relationships when the screw is fully screwed into the case, but only indirectly inferable by virtue of the user ending their screwing action.

5.2 Data collection protocol

The study was performed in a laboratory setting, where we set up workstations for each of the eight tasks described in the previous section. There was one workstation dedicated to each of the five non-mocktail tasks, and a sixth workstation for all three mocktail tasks; this latter workstation was equipped with a super-set of the necessary tools and ingredients for all mocktail recipes. The images at the beginning of each task recipe in Appendix A4 illustrate the various workstation setups.

We recruited participants for the study from among the co-workers at our organization, via broadly reaching emails and word-of-mouth. In total 21 participants engaged in the data collection study and provided permission for public data release. Each data collection session lasted up to ninety minutes, and participants were compensated 75 USD for their time and participation.

We used the following protocol with each participant in the data collection study. Upon arriving at the lab, the participant was greeted by the researcher running the study, and a brief introduction to the overall goals of the study was presented. After signing an informed consent and data release form, the participant watched a 5-minute slide deck to familiarize themselves with the HoloLens device and how the SIGMA system works; the deck included a short video with a first-person view of SIGMA

in action (on a different task than the ones used throughout the study). Next, the headset was placed on the participant, and an eye calibration process was performed. The participant was then given a final set of instructions: they were asked to do their best to complete each task, but were also informed that they could abandon a task at any point they felt unable to make progress or started to experience physical or mental discomfort; they were instructed not to interact with the researcher conducting the study; finally, they received a set of safety guidelines.

Following this briefing, each participant engaged in performing up to 6 tasks chosen from a broader set of predefined procedural tasks. The tasks were randomly chosen for each participant, following a stratified approach: a random order was created between the 6 workstations in the lab corresponding to the 5 non-mocktail tasks and the general mocktail category, and the choice of which mocktail to perform was also randomly chosen.

Throughout the task execution, the researcher conducting the study was separated visually (behind a panel) from the participant, but was able to monitor the interaction remotely, through live visualization of the data streams from the device enabled by the PsiStudio tool [7, 8]. In case of application crashes (a few happened occasionally), the experimenter intervened to inform the participant about the application failure and instruct them to move on to the next task. In addition, the experimenter intervened and ended the task in situations where it was clear that the participant was completely blocked from making further progress after committing an unrecoverable mistake. For example, the experimenter intervened if the participant accidentally triggered the Nespresso machine’s cleaning cycle, broke their pin-back button after pressing it incorrectly, started installing their skateboard components to the completely wrong part of the deck, and so on.

After each task, the experimenter decided if enough time was left to perform the next task. If so, the participant was instructed to proceed with the next task. If not, the experiment was brought to a conclusion and the participant was debriefed and asked to fill in a demographics questionnaire (shown in Appendix A5).

5.3 Demographic information

Of the 21 participants, 12 were male and 9 female. Additional demographic information from the post-experiment questionnaire is shown in Figure 4. Most participants were in the 46-55 age bracket (Figure 4.A). While their level of familiarity with AR/VR technologies varied, most participants had already en-

countered these technologies (Figure 4.B), even if they were not using them often (Figure 4.C)

5.4 Dataset statistics

We denote by *task execution session* or in short *session* a participant’s attempt to execute one of the tasks, regardless of whether the task was performed correctly. The 21 participants attempted a total of 95 task execution sessions. Of these, 10 sessions were excluded due to application crashes and performance issues. The SIGMACOLLAB dataset therefore consists of the remaining 85 task execution sessions. Together, these sessions span a total duration of 13 hours, 45 minutes and 11 seconds. Throughout the 85 sessions, the participants attempted 1583 sub-step executions and produced 3296 utterances.

Several additional statistics are presented in Table 3, and illustrated in Figure 5. The attempted task executions varied significantly in duration: the shortest attempted task execution in the dataset was 2 minutes and 1 second, and the longest 26 minutes and 11 seconds; the distribution of task durations is shown in Figure 5.C. We eliminated from the analysis sub-step executions in which the participant decided to abandon the task, or where the experimenter interrupted to stop the experiment, as described in the previous sub-section. The remaining 1570 attempted sub-step executions vary in duration, ranging from 2.16 seconds to 7 minutes and 53 seconds, with an average of 28.6 seconds; the distribution of sub-step durations is shown in Figure 5.D.

5.5 Expert Demonstrations

In addition to the 85 task execution sessions collected from the 21 participants, we also collected 8 *expert demonstration* sessions, one for each type of task, performed by a user who was very familiar with the tasks (one of the authors). During these expert demonstration sessions the user performed each step correctly, at a regular pace, and without asking any questions. As expected, these sessions are all successful and their durations are generally shorter than the corresponding task execution sessions from the participant data (as seen in Table 2 for expert demonstrations and Table 3 for participant data). The expert demonstration sessions are released together with the main dataset, and can be used in one-shot learning scenarios, or can provide useful prior information for a variety of downstream tasks and challenges.

5.6 Dataset Splits

The dataset is primarily intended to serve as a testbed for evaluating the generalizability and effectiveness of various models in the context of an interactive application, which is potentially out-of-distribution from what the models have been trained on. As such, we do not define specific train/val/test splits. That being said, specific train/test/val splits can be created based on the problem space investigated. When generating such splits, care should be exerted around stratification: the dataset can be stratified based on various data attributes, including but not limited to dataset sections (see Appendix A3), participant ids, task types, step types, task success, and combinations thereof.

6 ANNOTATIONS AND POST-PROCESSING

Next, we describe the set of manual annotations and additional post-processing that we performed on the collected data.

6.1 Manual Transcriptions

Throughout the data collection, SIGMA used the Silero [36] voice activity detector in conjunction with the `tiny.en` version of the Whisper [31] speech recognizer. Since both these models sometimes made errors, we manually segmented and transcribed the utterances spoken by the user, and provide the reference transcripts as part of the dataset release.

To assess the runtime performance of the Silero voice activity detector, we computed the detection error rate, *i.e.* the duration of falsely detected speech segments plus the duration of missed speech segments, divided by the total duration of the reference (manually transcribed) speech segments. The results indicate an average session-level detection error rate of 12.4%. To assess the runtime performance of the Whisper speech recognizer, we computed session-level word-error-rate by concatenating all the speech recognition results and all the manual utterance transcriptions from a session and computing word-error-rate on the resulting strings. The average session-level word-error-rate was 20.2%. These numbers confirm that the runtime speech recognition results contain significant errors, and the manual transcripts can provide an important resource for various downstream challenges and analyses.

6.2 Word-Level Timestamps

Whisper can be configured to provide word-level timestamps. However, our initial experimentation revealed that these timestamps are often inaccurate. We therefore computed word-level timings in post-processing, by using a force-alignment algorithm [2, 24]. The word-level timings were computed for:

- The runtime user utterances detected live, based on the Silero voice activity segments and corresponding Whisper speech recognition results;
- The manually-transcribed user utterances, based on the manual segmentations and corresponding user audio.
- The runtime system utterances, based on the synthesis text and corresponding synthesized audio segments;

The word-level timestamps can provide useful information for a variety of streaming tasks and challenges.

6.3 Task Success

We manually classified each task execution session into one of four different categories: *correctly-completed*, *incorrectly-completed*, *abandoned* and *system-failure*. A task execution session was labeled *correctly-completed* if the participant reached the correct world state at the end of the task *i.e.*, the same world state the expert reached during the expert demonstration session. Note that it is possible that in *correctly-completed* sessions some of the sub-steps were done incorrectly, but either redone at a later stage or recovered via some alternative method, in such a way that at the end, when the participant indicated they are done with the last step of the procedure, the correct end state was reached.

Task	Task Execution Sessions (#)	Total Duration (hh:mm:ss)	Sub-step Executions (#)	User Utterance (#)	Avg. Duration (mm:ss)
Nespresso	17	01:05:16	148	295	03:50
Hard-drive	15	01:12:41	178	288	04:51
Skateboard	10	02:25:20	218	540	14:32
Button	12	02:36:33	296	619	13:03
Notebook	15	03:35:37	480	943	14:22
Mocktail	16	02:49:45	263	611	10:36
Margarita	5	00:47:27	72	161	09:29
Mojito	6	01:01:05	99	222	10:11
Whiskey	5	01:01:13	92	228	12:15
Overall (Dataset)	85	13:45:11	1583	3296	09:46

Table 3: Dataset statistics, overall and by task type.

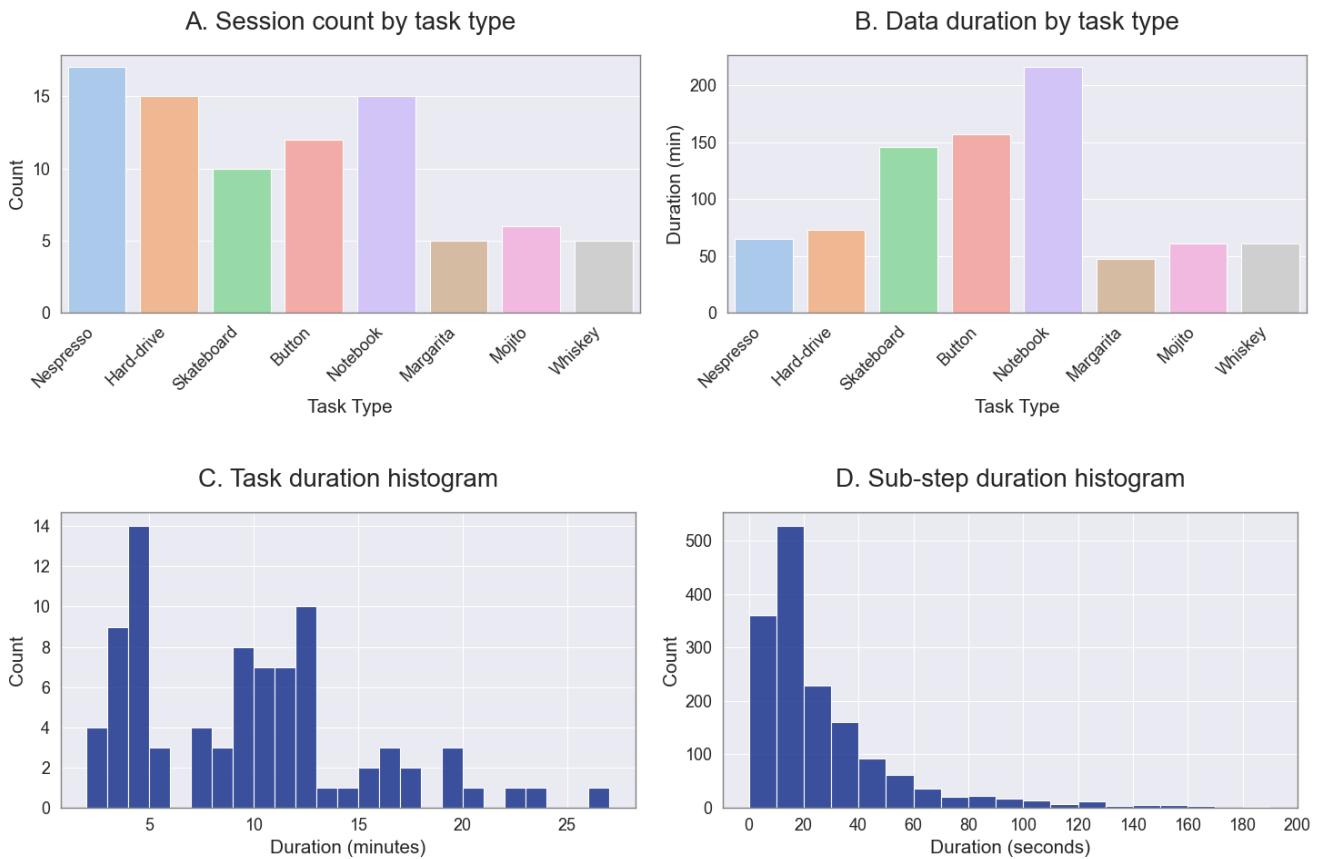


Figure 5: Dataset statistics.

A task execution session was labeled *incorrectly-completed* if the participant reached the end of the task by advancing through the steps, but did not reach the correct world state at the end of the task; for instance, this can happen if some of the sub-steps are not performed correctly, in a manner that leads to an incorrect result at the end of the task. Sessions where the participant decided to stop before reaching the last step were labeled as *abandoned*. Finally, sessions in which the system failed in a manner that led to an early stop of the interaction

(e.g., crashes, significant performance degradation observed by the experimenter, etc.) were labeled as *system-failure*.

We computed *task success rate* by eliminating the *system-failure* sessions and then dividing the number of correctly-completed sessions by the sum of correctly-completed, incorrectly-completed and abandoned sessions. The counts of task execution sessions falling into each of the categories described above and the task success rates are reported on the last column in Table 4.

Task	Total Sessions (#)	Correctly Completed (#)	Incorrectly Completed (#)	Abandoned (#)	System Failure (#)	Task Success Rate (%)
Nespresso	17	13	0	4	0	76.5%
Hard-drive	15	14	0	0	1	100.0%
Skateboard	10	9	0	1	0	90.0%
Button	12	7	0	3	2	70.0%
Notebook	15	7	3	3	2	53.8%
Mocktail	16	10	5	1	0	62.5%
Margarita	5	5	0	0	0	100.0%
Mojito	6	3	2	1	0	50.0%
Whiskey	5	2	3	0	0	40.0%
Overall (Dataset)	85	60	8	12	5	75.0%

Table 4: Classification of task execution sessions and task success rates.

6.4 Gaze Signal Post-Processing

Gaze behavior is a rich and powerful signal for inferring various aspects of the user’s cognitive state, such as confidence, confusion, distraction, frustration, flow state, and so on [35]. To better support such modeling and inference, we performed some additional post-processing on the gaze signal.

First, we generated an annotation stream marking the temporal periods when the participant was likely reading the instructions displayed in the virtual task panel, rather than looking at the physical task space (the intervals are illustrated in Figure 3.K). We generated these annotations automatically by intersecting the user’s gaze ray with the location of the floating user interface rectangle holograms in 3D space (see Figure 3.B). To smooth these annotations, gaze-to-interface periods shorter than 0.5 seconds were eliminated, and annotations with intervening gaps of less than 0.5 seconds were merged together.

In addition, we also computed the intersection of the gaze ray with the 3D depth map, then projected that 3D intersection point onto the color image, both (left and right) grayscale image views, and depth image. This gaze point projection was accomplished by leveraging the camera extrinsics and intrinsics information captured in the data. The resulting data streams contain the pixel coordinates for the gaze point in each of the corresponding image spaces.

7 CONCLUSION

We have introduced SIGMACOLLAB, an interactive, application-driven dataset that aims to support research on various challenges related to real-time collaboration in physically situated settings. The dataset contains approximately 14 hours of interaction data in which untrained participants were guided by a mixed-reality AI assistive application through performing a variety of procedural tasks. SIGMACOLLAB subsumes a rich set of multimodal data streams, including egocentric views from color, grayscale and depth cameras, head, hand and gaze tracking information, as well as additional annotations, such as manual transcripts and word-level timings for the spoken utterances.

We believe that an application-driven approach to dataset construction confers important properties for identifying and studying a variety of challenges related to interaction, coordination and collaboration in physically situated settings. In future work,

we plan to use the data to establish a set of new benchmarks that complement preexisting egocentric computer vision benchmarks by more closely focusing on interaction-related challenges, such as timing, proactive interventions, grounding, detecting user cognitive states such as frustration and confusion, *etc.* We hope the community joins us in adopting this new resource, together with the SIGMA open-source application that was used to collect it, and leverages it to make progress towards seamless collaboration between humans and machines in the physical world.

REFERENCES

- [1] Azure ai speech. <https://azure.microsoft.com/en-us/products/ai-services/ai-speech>. Accessed on Sept 27th, 2023.
- [2] Forced alignment with wav2vec2. https://docs.pytorch.org/audio/stable/tutorials/forced_alignment_tutorial.html. Accessed on May 5th, 2025.
- [3] Sigma: Situated interaction guidance monitoring and assistance. <https://github.com/microsoft/psi/blob/master/Applications/Sigma/Readme.md>. Accessed on April 28th, 2025.
- [4] S. Bansal, C. Arora, and C. Jawahar. My view is the best view: Procedure learning from egocentric videos. In *European Conference on Computer Vision (ECCV)*, 2022.
- [5] Y. Bao, K. Yu, Y. Zhang, S. Storks, I. Bar-Yossef, A. de la Iglesia, M. Su, X. Zheng, and J. Chai. Can foundation models watch, talk and guide you step by step to make a cake? In H. Bouamor, J. Pino, and K. Bali, eds., *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 12325–12341. Association for Computational Linguistics, Singapore, Dec. 2023. doi: 10.18653/v1/2023.findings-emnlp.824
- [6] D. Bohus, S. Andrist, Y. Bao, E. Horvitz, and A. Paradiso. Is this it?: Towards ecologically valid benchmarks for situated collaboration. In *Companion Proceedings of the 26th International Conference on Multimodal Interaction*, pp. 41–45, 2024.
- [7] D. Bohus, S. Andrist, A. Feniello, N. Saw, M. Jalobeanu, P. Sweeney, A. L. Thompson, and E. Horvitz. Platform for situated intelligence, 2021.

- [8] D. Bohus, S. Andrist, and M. Jalobeanu. Rapid development of multimodal interactive systems: a demonstration of platform for situated intelligence. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 493–494, 2017.
- [9] D. Bohus, S. Andrist, N. Saw, A. Paradiso, I. Chakraborty, and M. Rad. Sigma: An open-source interactive system for mixed-reality task assistance research, 2024.
- [10] D. Bohus, S. Andrist, N. Saw, A. Paradiso, I. Chakraborty, and M. Rad. Sigma: An open-source interactive system for mixed-reality task assistance research – extended abstract. In *2024 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*, pp. 889–890, 2024. doi: 10.1109/VRW62533.2024.00241
- [11] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- [12] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6299–6308, 2017.
- [13] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3213–3223, 2016.
- [14] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018.
- [15] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. The epic-kitchens dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 43(11):4125–4141, 2021. doi: 10.1109/TPAMI.2020.2991965
- [16] D. Damen, H. Doughty, G. M. Farinella, A. Furnari, J. Ma, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision (IJCV)*, 130:33–55, 2022.
- [17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.
- [18] A. Fathi, Y. Li, and J. M. Rehg. Learning to recognize daily actions using gaze. In *European Conference on Computer Vision*, 2012.
- [19] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pp. 3354–3361. IEEE, 2012.
- [20] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, M. Martin, T. Nagarajan, I. Radosavovic, S. K. Ramakrishnan, F. Ryan, J. Sharma, M. Wray, M. Xu, E. Z. Xu, C. Zhao, S. Bansal, D. Batra, V. Cartillier, S. Crane, T. Do, M. Doulaty, A. Erappalli, C. Feichtenhofer, A. Fragomeni, Q. Fu, C. Fuegen, A. Gebreselasie, C. Gonzalez, J. Hillis, X. Huang, Y. Huang, W. Jia, W. Khoo, J. Kolar, S. Kottur, A. Kumar, F. Landini, C. Li, Y. Li, Z. Li, K. Mangalam, R. Modhugu, J. Munro, T. Murrell, T. Nishiyasu, W. Price, P. R. Puentes, M. Ramazanova, L. Sari, K. Somasundaram, A. Southerland, Y. Sugano, R. Tao, M. Vo, Y. Wang, X. Wu, T. Yagi, Y. Zhu, P. Arbelaez, D. Crandall, D. Damen, G. M. Farinella, B. Ghanem, V. K. Ithapu, C. V. Jawahar, H. Joo, K. Kitani, H. Li, R. Newcombe, A. Oliva, H. S. Park, J. M. Rehg, Y. Sato, J. Shi, M. Z. Shou, A. Torralba, L. Torresani, M. Yan, and J. Malik. Ego4d: Around the World in 3,000 Hours of Egocentric Video. In *IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*, 2022.
- [21] K. Grauman, A. Westbury, L. Torresani, K. Kitani, J. Malik, T. Afouras, K. Ashutosh, V. Baiyya, S. Bansal, B. Boote, E. Byrne, Z. Chavis, J. Chen, F. Cheng, F.-J. Chu, S. Crane, A. Dasgupta, J. Dong, M. Escobar, C. Forigua, A. Gebreselasie, S. Haresh, J. Huang, M. M. Islam, S. Jain, R. Khirodkar, D. Kukreja, K. J. Liang, J.-W. Liu, S. Majumder, Y. Mao, M. Martin, E. Mavroudi, T. Nagarajan, F. Ragusa, S. K. Ramakrishnan, L. Seminara, A. Somayazulu, Y. Song, S. Su, Z. Xue, E. Zhang, J. Zhang, A. Castillo, C. Chen, X. Fu, R. Furuta, C. Gonzalez, P. Gupta, J. Hu, Y. Huang, Y. Huang, W. Khoo, A. Kumar, R. Kuo, S. Lakhavani, M. Liu, M. Luo, Z. Luo, B. Meredith, A. Miller, O. Oguntola, X. Pan, P. Peng, S. Pramanick, M. Ramazanova, F. Ryan, W. Shan, K. Somasundaram, C. Song, A. Southerland, M. Tateno, H. Wang, Y. Wang, T. Yagi, M. Yan, X. Yang, Z. Yu, S. C. Zha, C. Zhao, Z. Zhao, Z. Zhu, J. Zhuo, P. Arbelaez, G. Bertasius, D. Crandall, D. Damen, J. Engel, G. M. Farinella, A. Furnari, B. Ghanem, J. Hoffman, C. V. Jawahar, R. Newcombe, H. S. Park, J. M. Rehg, Y. Sato, M. Savva, J. Shi, M. Z. Shou, and M. Wray. Ego-exo4d: Understanding skilled human activity from first- and third-person perspectives, 2023.
- [22] R. Hazra, B. Chen, A. Rai, N. Kamra, and R. Desai. Egotv: Egocentric task verification from natural language task descriptions, 2023.
- [23] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. In *Workshop on faces in 'Real-Life' Images: detection, alignment, and recognition*, 2008.
- [24] L. Kürzinger, D. Winkelbauer, L. Li, T. Watzel, and G. Rigoll. *CTC-Segmentation of Large Corpora for German End-to-End Speech Recognition*, p. 267–278. Springer International Publishing, 2020. doi: 10.1007/978-3-030-60276-5_27
- [25] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

- [26] Y. Li, M. Liu, and J. M. Rehg. In the eye of the beholder: Gaze and actions in first person video, 2020.
- [27] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.
- [28] A. Majumdar, A. Ajay, X. Zhang, P. Putta, S. Yenamandra, M. Henaff, S. Silwal, P. Mcvay, O. Maksymets, S. Arnaud, K. Yadav, Q. Li, B. Newman, M. Sharma, V. Berges, S. Zhang, P. Agrawal, Y. Bisk, D. Batra, M. Kalakrishnan, F. Meier, C. Paxton, S. Sax, and A. Rajeswaran. Openeqa: Embodied question answering in the era of foundation models. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [29] A. Miech, D. Zhukov, J.-B. Alayrac, M. Tapaswi, I. Laptev, and J. Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips, 2019.
- [30] R. Peddi, S. Arya, B. Challa, L. Pallapothula, A. Vyas, B. Gouripeddi, J. Wang, Q. Zhang, V. Komaragiri, E. Ragan, N. Ruozzi, Y. Xiang, and V. Gogate. Captaincook4d: A dataset for understanding errors in procedural activities, 2024.
- [31] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pp. 28492–28518. PMLR, 2023.
- [32] T. J. Schoonbeek, T. Houben, H. Onvlee, P. H. N. de With, and F. van der Sommen. Industreal: A dataset for procedure step recognition handling execution errors in egocentric videos in an industrial-like setting, 2023.
- [33] F. Sener, D. Chatterjee, D. Shelepor, K. He, D. Singhania, R. Wang, and A. Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. *CVPR 2022*.
- [34] F. Sener, D. Chatterjee, D. Shelepor, K. He, D. Singhania, R. Wang, and A. Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21096–21106, June 2022.
- [35] M. Stiber, D. Bohus, and S. Andrist. "uh, this one?": Leveraging behavioral signals for detecting confusion during physical tasks. In *Proceedings of the 26th International Conference on Multimodal Interaction*, pp. 194–203, 2024.
- [36] S. Team. Silero vad: pre-trained enterprise-grade voice activity detector (vad), number detector and language classifier. <https://github.com/snakers4/silero-vad>, 2024.
- [37] D. Ungureanu, F. Bogo, S. Galliani, P. Sama, X. Duan, C. Meekhof, J. Stühmer, T. J. Cashman, B. Tekin, J. L. Schönberger, P. Olszta, and M. Pollefeys. Hololens 2 research mode as a tool for computer vision research. *CoRR*, abs/2008.11239, 2020.
- [38] X. Wang, T. Kwon, M. Rad, B. Pan, I. Chakraborty, S. Andrist, D. Bohus, A. Feniello, B. Tekin, F. V. Frumeri, N. Joshi, and M. Pollefeys. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20270–20281, October 2023.
- [39] L. Zhou, C. Xu, and J. J. Corso. Towards automatic learning of procedures from web instructional videos, 2017.

APPENDIX

A1 A SAMPLE INTERACTION WITH SIGMA

Below we present the transcript of an interaction between SIGMA and one of the participants, denoted by **User**.

- +00:03.9 SIGMA: Hi! This is Sigma, your mixed-reality task assistant.
- +00:08.9 SIGMA: Today I'm here to help you make a button using a button press machine.
- +00:12.6 SIGMA: You can move me anytime to a more convenient location by pinching and dragging the blue diamond gem.
- +00:18.4 SIGMA: Let me know when you're ready to start.
- +00:23.7 **User:** **i'm ready**
- +00:25.7 SIGMA: The first step is to setup the cutting system.
- +00:28.8 SIGMA: First, attach the white top cutting insert into the top housing. The top cutting insert is the smaller white piece with the notches. Hold it with the notches up and align one of the notches to the arrow on the top housing and press up to magnetize.
- +00:45.3 **User:** **next step**
- +00:46.9 SIGMA: Next, place the base cutting insert into the base housing, with the flat face up. The base cutting insert is the bigger white piece.
- +00:57.8 **User:** **hmm**
- +01:02.9 **User:** **does this look right**
- +01:07.3 SIGMA: Yes, the base cutting insert looks correctly placed with the flat face up in the base housing.
- +01:13.1 **User:** **great next step**
- +01:14.9 SIGMA: Great.
- +01:16.0 SIGMA: The next step is to prepare the design for cutting.
- +01:19.3 SIGMA: First, place the paper with the design on the white base.
- +01:23.9 **User:** **hmm oh boy this is very cute**
- +01:28.0 **User:** **um**
- +01:32.2 **User:** **i might choose this one**
- +01:36.5 **User:** **next step**
- +01:38.3 SIGMA: Next, place the cutting die on top of the paper. The cutting die is the teal piece with a hole in it. Place it over the paper, with the black side facing down, in contact with the paper.
- +01:52.8 **User:** **should i be trying to center it in any specific way is the inside of the hole what's gonna be cut out**
- +02:01.9 SIGMA: Yes, you should center the design so that the part visible through the hole in the cutting die is what you want on the button. make sure the cross lines on the cutting die align with the horizontal and vertical axes of your design.
- +02:08.3 **User:** **do i need to compen-**
- +02:14.9 **User:** **do i need to compensate for the curvature of the button when i pick where i want to center this**
- +02:23.9 SIGMA: No, you don't need to compensate for the curvature of the button; just center the design as you want it to appear on the flat surface.
- +02:31.7 **User:** **okay next step**
- +02:34.1 SIGMA: Next, if you look from the top at the paper, you will see how the button will look like through the center hole of the cutting die. Make sure things are aligned so the design is centered and the cross lines on top of the cutting die correspond to the horizontal and vertical axes of the design.
- +02:49.5 **User:** **next step**
- +02:51.2 SIGMA: Next, swivel the base housing with the design and cutting die under the button press by rotating the whole base clockwise around the metal leg. It should click into place.
- +03:06.3 **User:** **okay**
- +03:08.5 **User:** **next step**
- +03:10.2 SIGMA: Finally, if the cutting die has slightly shifted position adjust it so it's relatively centered under the press.
- +03:20.0 **User:** **okay next step**
- +03:22.6 SIGMA: Great.
- +03:23.9 SIGMA: The next step is to cut the design.
- +03:26.4 SIGMA: First, push the top handle down and apply pressure to cut the design. You will need to apply significant pressure down, from a standing position, with both hands on the handle, from the top. You should hear or feel a faint click as the design is being cut, but it may not feel obvious.
- +03:59.8 **User:** **okay next step**

- +04:01.7 SIGMA: Next, swivel the base housing back out counterclockwise and remove the teal cutting die and design from the base.
- +04:15.1 User: **next step**
- +04:16.7 SIGMA: Next, take a look at the design, and, if it was not completely cut, use the scissors to finish the circular cut.
- +04:24.4 User: **it looks like i put my design a little too high and i don't have a complete circle i have a flat top should i pick a new one or can i still use this**
- +04:37.4 SIGMA: If the flat top cuts off an important part of the design, it's best to pick a new one and try again, ensuring the design is centered properly.
- +04:47.1 User: **it isn't an important part**
- +04:51.6 SIGMA: Ready for the next step?
- +04:53.6 User: **sure let's do it**
- +04:55.6 SIGMA: Next, remove the white base cutting insert from the base.
- +05:00.5 User: **next step**
- +05:02.1 SIGMA: Finally, remove the white top cutting insert from the press head.
- +05:07.7 User: **next step**
- +05:09.5 SIGMA: Great.
- +05:10.7 SIGMA: The next step is to setup the button press.
- +05:13.8 SIGMA: First, attach the top insert to the top housing. The top insert is the teal cylindrical piece with small notches. Hold it with the notches up, align the notch marked with the letter 'B' with the arrow on the top housing and press upward to magnetize. The outer ring will
- +05:29.7 User: **hmm**
- +05:44.4 User: **hmm**
- +05:50.7 User: **okay next step**
- +05:53.1 SIGMA: Next, place the base insert on the base housing and rotate until the piece clicks and locks into place. The base insert is the teal cylindrical piece with large notches.
- +06:11.1 User: **next step**
- +06:12.8 SIGMA: Great.
- +06:14.0 SIGMA: The next step is to load the button into the press.
- +06:17.2 SIGMA: First, rotate both top and base inserts into position marked with the letter 'A', aligning with arrows on the top and bottom housing.
- +06:25.9 User: **next step**
- +06:27.6 SIGMA: Next, place a rounded metal button shell with the rounded face up into the base insert.
- +06:45.9 User: **next step**
- +06:47.6 SIGMA: Next, layer the cut paper design over the button shell. Use the pin line as a guide to make sure the text and imagery is straight.
- +07:03.5 User: **next step**
- +07:05.3 SIGMA: Next, take a mylar piece out of the teal envelope and place it over the cut paper design.
- +07:23.2 User: **next step**
- +07:24.8 SIGMA: Next, swivel the base housing into place below the button press by rotating clockwise. It should click into place.
- +07:40.8 User: **next step**
- +07:42.5 SIGMA: Next, press the handle once. This action will capture the button in the top insert.
- +07:49.7 User: **do i need to press hard**
- +07:54.3 User: **oh next step**
- +07:57.7 SIGMA: Finally, swivel the base back out, counterclockwise. Note: the base should be empty now, the top pieces of the button are inside the top insert.
- +08:07.7 User: **next step**
- +08:09.6 SIGMA: Great.
- +08:10.8 SIGMA: The next step is to make the button.
- +08:13.1 SIGMA: First, rotate both the top and base inserts to the position marked with the letter 'B', aligning with the corresponding arrows on the top and bottom housing.
- +08:22.6 User: **next step**
- +08:26.0 SIGMA: Next, place the metal pin backing with the pin side down and the zig-zag part up into the base insert.
- +08:38.7 User: **next step**
- +08:40.8 SIGMA: Next, align the pin parallel to the pin line and check that the zig-zag part of the pin is sitting down flush with the metal.
- +09:02.3 User: **next step**
- +09:03.8 SIGMA: Next, swivel the base housing into place below the button press by rotating clockwise. It should click into place.

+09:18.0 **User:** **next step**
+09:20.1 SIGMA: Next, press the handle.
+09:29.4 **User:** **next step**
+09:31.2 SIGMA: Finally, swivel the base housing back out by rotating counterclockwise, and retrieve the finished button.
+09:41.7 **User:** **next step**
+09:43.6 SIGMA: Congratulations. You have finished the task.
+09:47.7 SIGMA: I'll see you next time. Bye bye.
+09:49.6 **User:** **thank you**

A2 RESPONSE-GENERATING PROMPT

Below is a sample response-generating prompt that SIGMA used at the end of the “*Does this look great?*” runtime-detected user utterance³ in the sample interaction illustrated in Figure 1 and Appendix A1.

The prompt template is shown in regular black typeface, and various pieces of context that are automatically populated in the prompt template are shown in different colors:

- **blue**: the task name
- **dark green**: the task recipe
- **dark cyan**: the current step the user is on
- **brown**: the current sub-step the user is on
- **orange**: the history of the dialog in the sub-step
- **purple**: the last speech recognition result

In addition, an image collected from the right grayscale camera at the time the user started speaking their utterance is also injected into the prompt template, as shown below.

PROMPT:

You are a helpful assistant guiding a user via spoken interaction to perform the following task in the physical world:
Make a button using a button press machine

You are helping the user to perform this task based on the following task recipe:

Make a button using a button press machine

1. Setup the cutting system.

1.1. Attach the white **top cutting insert** into the **top housing**. The **top cutting insert** is the smaller white piece with the notches. Hold it with the notches up and align one of the notches to the arrow on the **top housing** and press up to magnetize.

1.2. Place the **base cutting insert** into the **base housing**, with the flat face up. The **base cutting insert** is the bigger white piece.

2. Prepare the design for cutting.

2.1. Place the paper with the design on the white base.

2.2. Place the **cutting die** on top of the paper. The **cutting die** is the teal piece with a hole in it. Place it over the paper, with the black side facing down, in contact with the paper.

[...]⁴

The user is on the following step: 1. Setup the cutting system

The user is on the following substep: 1.2. Place the **base cutting insert** into the **base housing**, with the flat face up. The **base cutting insert** is the bigger white piece.

Here is a transcript of the dialog since the instructions for the last substep were delivered to the user, where the timings of each utterance are shown at the beginning of each utterance, in seconds, relative to time 0, when the instruction was first delivered:

[+00:00.0] You(Assistant): Next, place the base cutting insert into the base housing, with the flat face up. The base cutting insert is the bigger white piece.

[+00:11.1] User: Hmm.

[+00:16.2] User: Does this look great?

The speech recognition result for the last user utterance was: " Does this look great?"

The utterance spoken by the user is probably close to this speech recognition results, but there might be some slight differences because of speech recognition errors.

For additional context, here is also the image that the user saw when they started speaking their utterance:

³Note that the actual utterance spoken by the user, as shown in Appendix A1 is “Does this look right?”; in this case the system committed a speech-recognition error at runtime.

⁴the reminder of the recipe is omitted for space reasons; the full task recipe is available in Appendix A4



Based on the dialog context and speech recognition results, and using information from image as context, please classify the LAST user utterance in one of four classes: question, no-question, next-step, or step-navigation. If you think the utterance is a question, also provide an appropriate answer.

Please provide your response using precisely the following json format:

```
{
    "classification": "question",
    "answer": "The apple is on the red box"
}
```

Here is more information on how to perform the classification:

- If the user utterance is a question related to this step or substep, then classify the utterance as "question", place your answer to the question in the answer field. Please provide a concise answer, as the answer will be spoken back to the user.
- If the user utterance indicates that they want to move to the next substep, or if it describes a state of affairs where they have completed the substep then classify as "next-step" and leave the answer field blank.
- If the user is issuing a navigational command to go to a different step, substep or step.substep combination, or if they are signaling they are already on a different step or substep, then classify the utterance as "step-navigation" and provide the target step and substep in the form "step=#,substep=#" in the answer. If only a step number is provided by the user, answer in the form "step=#".
- If the user looks like they are talking or making a comment to themselves, or if it looks like they are repeating the instruction (or a portion of it) to themselves, then classify as "no-question" and leave the answer field blank.
- If you are unsure or in all other cases, please classify the utterance as "no-question" and leave the answer field blank.

Provide your response exactly matching the JSON format described above and make sure you provide a well-formatted json. Do not provide any text before or after the json, just start with an open brace and end with a closing brace.

MODEL RESPONSE:

```
{
    "classification": "question",
    "answer": "Yes, the base cutting insert looks correctly placed with the flat face up in the base housing."
}
```

A3 DATASET SECTIONS

Three different Azure GPT model deployments were used in conjunction with the response-generating prompt throughout the data collection study. In the first part of the study we relied on a GPT-4o model using a global data zone region. After observing relatively large response latencies, we switched to a GPT-4o-mini deployment in a US data zone region. Finally, for the last third of the dataset, we switched back to a GPT-4o deployment using a US data zone region.

Because of these changes, these three sections of the dataset — which we refer to as Section A, B, and C — have slightly different characteristics in terms of latency and GPT responses to user queries. Table A1 provides the details regarding which task execution sessions belong in each section, and Table A2 shows the number of task execution sessions for each task type across the three dataset sections.

Section	GPT model	From Session	To Session	Sessions Count
A	GPT-4o/Global Data Zone	20250228-134800	20250307-102820	28
B	GPT-4o-mini/US Data Zone	20250318-154110	20250324-105658	27
C	GPT-4o/US Data Zone	20250325-110228	20250417-160112	30

Table A1: Dataset Sections.

Task	Sessions in Section A (#)	Sessions in Section B (#)	Sessions in Section C (#)	Sessions Total (#)
Nespresso	6	5	6	17
Hard-drive	5	6	4	15
Skateboard	3	3	4	10
Button	5	3	4	12
Notebook	5	5	5	15
Mocktail	4	5	7	16
Margarita	2	2	1	5
Mojito	1	2	3	6
Whiskey	1	1	3	5
Overall (Dataset)	85	60	8	12

Table A2: Task execution counts by task type in each dataset section.

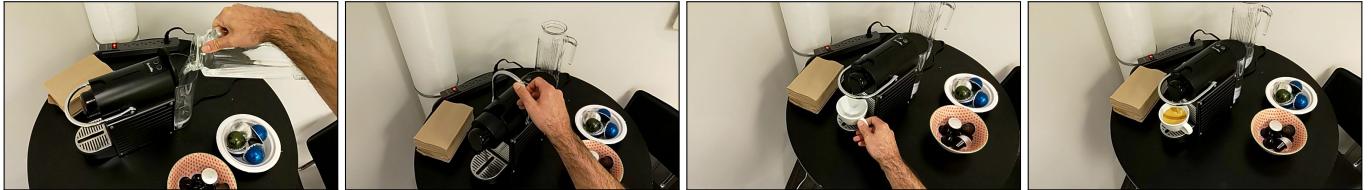
A4 TASK RECIPES

Below are the recipes for the 8 tasks used in the SIGMACOLLAB dataset. For each task, four images sampled from a reference execution of the task are also shown for illustration purposes. The classification of the sub-step per the taxonomy described in Section 5.1 is also shown after each sub-step instructions, in bracketed light gray text.

Task 1: Make a Mojito mocktail



1. Muddle the mint
 - 1.1. Add 7 mint leaves in the larger tin of a Boston shaker. [Add Amount]
 - 1.2. Add a teaspoon of soda water. [Add Amount]
 - 1.3. Add half a teaspoon of sugar. [Add Amount]
 - 1.4. Muddle the mint by repeatedly hammering it in the bottom of the tin with the flat end of the muddler. [Invisible State]
2. Prepare the ingredients
 - 2.1. Using the citrus squeezer, extract the juice from two lime halves into a cup. [Add Amount]
 - 2.2. Using the jigger, measure and pour 3/4 ounces of fresh lime juice into the tin with the mint. [Add Amount]
 - 2.3. Pour 2 ounces of Seedlip Garden non-alcoholic gin into the tin. [Add Amount]
 - 2.4. Pour in half an ounce of simple syrup. [Add Amount]
 - 2.5. Add 6 or 7 ice-cubes into the tin. [Add Amount]
3. Shake the cocktail
 - 3.1. Cover the large tin of the Boston shaker with the smaller tin, and push the top (smaller) tin down firmly to ensure a seal. [Invisible State]
 - 3.2. Using both hands to catch the tins, make sure they stay pressed together and shake the drink 7 to ten seconds, until the tins become cold to the touch. Make sure you hold them tight, as the shaking process might have them come undone and the drink might spill. [Invisible State]
 - 3.3. Open the top tin. If it does not come out easily, you can tap it from the side, or try to rotate it up. [Move]
 - 3.4. Place a Hawthorne strainer over the opening of the large tin. [Move]
4. Pour the mocktail
 - 4.1. Add a mint sprig into a highball glass. [Discrete Event]
 - 4.2. Fill the glass three quarters with ice. [Add Amount]
 - 4.3. While holding the strainer over the top of the tin, tilt the tin and strain the drink into the prepared glass. [Add Amount]
 - 4.4. Top the glass with soda water. [Add Amount]

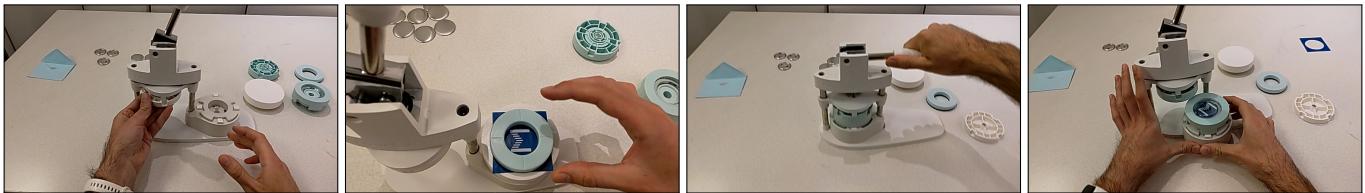
Task 2: Make coffee using a Nespresso Pixie machine

1. Fill the water tank with fresh water
 - 1.1. Open the water tank lid [Discrete Event]
 - 1.2. Fill the water tank with fresh water [Add Amount]
 - 1.3. Close the water tank lid [Discrete Event]
2. Turn on the machine
 - 2.1. Briefly push the two buttons on top of the machine simultaneously. The buttons should start blinking. [Discrete Event]
 - 2.2. Wait until the buttons stop blinking and the machine is ready for use [Discrete Event]
3. Insert a coffee capsule
 - 3.1. Lift the front lever upwards. [Discrete Event]
 - 3.2. Insert a Nespresso coffee capsule into the capsule compartment, with the flat side towards the front of the machine. [Discrete Event]
 - 3.3. Lower the lever to close the compartment. [Discrete Event]
4. Brew the coffee
 - 4.1. Place the coffee cup on the drip tray below the nozzle. [Move]
 - 4.2. Press the espresso or the lungo button and wait until the coffee stops brewing. [Discrete Event]

Task 3: Replace a hard-drive in a PC

1. Open the PC case
 - 1.1. To remove the case lid you will have to first unscrew the case screw using the screwdriver. The case screw is the single screw connecting the back of the PC to the black case lid. [Move]
 - 1.2. Pull the case-lid handle and open and remove the case lid. Place the lid on the table next to you. [Move]
2. Remove the existing hard-drive
 - 2.1. Look for two hard drives installed towards the front of the PC case. Disconnect the wide connector marked D2, from the hard drive that is closer to you. [Move]
 - 2.2. Disconnect the narrow connector. [Move]
 - 2.3. Lift the hard-drive out by pushing the two green tabs towards each other and pulling up. [Move]
 - 2.4. Remove the plastic casing from around the hard-drive. [Move]
3. Insert the new hard-drive
 - 3.1. Place the plastic casing around the new hard-drive. [Discrete Event]
 - 3.2. Insert the hard-drive into the bay. [Discrete Event]
 - 3.3. Plug the narrow connector into the hard-drive. [Discrete Event]
 - 3.4. Plug the wide connector into the hard-drive. [Discrete Event]
4. Close the PC case
 - 4.1. Slide the case lid in and close it down. [Discrete Event]
 - 4.2. Screw the case screw back using the screwdriver. [Invisible State]

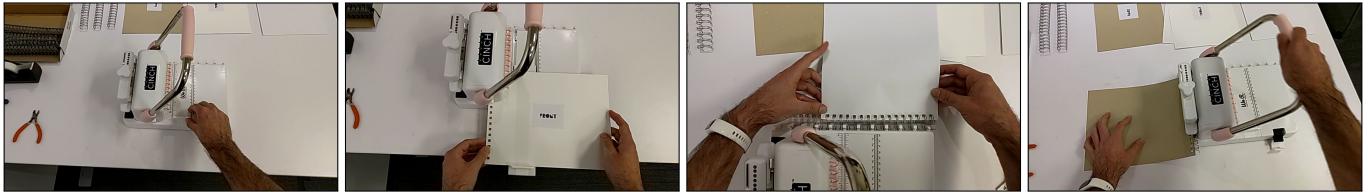
Task 4: Make a button using a button press machine



1. Setup the cutting system
 - 1.1. Attach the white top cutting insert into the top housing. The top cutting insert is the smaller white piece with the notches. Hold it with the notches up and align one of the notches to the arrow on the top housing and press up to magnetize. [Discrete Event]
 - 1.2. Place the base cutting insert into the base housing, with the flat face up. The base cutting insert is the bigger white piece. [Discrete Event]
2. Prepare the design for cutting
 - 2.1. Place the paper with the design on the white base. [Move]
 - 2.2. Place the cutting die on top of the paper. The cutting die is the teal piece with a hole in it. Place it over the paper, with the black side facing down, in contact with the paper. [Move]
 - 2.3. If you look from the top at the paper, you will see how the button will look like through the center hole of the cutting die. Make sure things are aligned so the design is centered and the cross lines on top of the cutting die correspond to the horizontal and vertical axes of the design. [Check]
 - 2.4. Swivel the base housing with the design and cutting die under the button press by rotating the whole base clockwise around the metal leg. It should click into place. [Discrete Event]
 - 2.5. If the cutting die has slightly shifted position adjust it so it's relatively centered under the press. [Check]
3. Cut the design
 - 3.1. Push the top handle down and apply pressure to cut the design. You will need to apply significant pressure down, from a standing position, with both hands on the handle, from the top. You should hear or feel a faint click as the design is being cut, but it may not feel obvious. [Invisible State]
 - 3.2. Swivel the base housing back out counterclockwise and remove the teal cutting die and design from the base. [Move]
 - 3.3. Take a look at the design, and, if it was not completely cut, use the scissors to finish the circular cut. [Check]
 - 3.4. Remove the white base cutting insert from the base. [Move]
 - 3.5. Remove the white top cutting insert from the press head. [Move]
4. Setup the button press
 - 4.1. Attach the top insert to the top housing. The top insert is the teal cylindrical piece with small notches. Hold it with the notches up, align the notch marked with the letter 'B' with the arrow on the top housing and press upward to magnetize. The outer ring will drop when the center is secure. [Discrete Event]
 - 4.2. Place the base insert on the base housing and rotate until the piece clicks and locks into place. The base insert is the teal cylindrical piece with large notches. [Discrete Event]
5. Load the button into the press
 - 5.1. Rotate both top and base inserts into position marked with the letter 'A', aligning with arrows on the top and bottom housing. [Discrete Event]
 - 5.2. Place a rounded metal button shell with the rounded face up into the base insert. [Move]
 - 5.3. Layer the cut paper design over the button shell. Use the pin line as a guide to make sure the text and imagery is straight. [Move]
 - 5.4. Take a mylar piece out of the teal envelope and place it over the cut paper design. [Move]
 - 5.5. Swivel the base housing into place below the button press by rotating clockwise. It should click into place. [Discrete Event]
 - 5.6. Press the handle once. This action will capture the button in the top insert. [Discrete Event]
 - 5.7. Swivel the base back out, counterclockwise. Note: the base should be empty now, the top pieces of the button are inside the top insert. [Move]
6. Make the button
 - 6.1. Rotate both the top and base inserts to the position marked with the letter 'B', aligning with the corresponding arrows on the top and bottom housing. [Discrete Event]

- 6.2. Place the metal pin backing with the pin side down and the zig-zag part up into the base insert. [Move]
- 6.3. Align the pin parallel to the pin line and check that the zig-zag part of the pin is sitting down flush with the metal. [Check]
- 6.4. Swivel the base housing into place below the button press by rotating clockwise. It should click into place. [Discrete Event]
- 6.5. Press the handle. [Discrete Event]
- 6.6. Swivel the base housing back out by rotating counterclockwise, and retrieve the finished button. [Move]

Task 5: Make a notebook using a binding machine



1. Prepare the binding machine
 - 1.1. Unhook the storage hook from the handle bar. The handle bar should automatically lift once the hook is released. [Move]
 - 1.2. Make sure all the pink pegs, from number 1 to 12 are pushed in. [Check]
 - 1.3. Make sure the edge guide is pushed all the way into the machine. The edge guide is the vertical plastic edge connected to the ruler. [Check]
2. Punch the short edge of the covers
 - 2.1. Take a front and back cover and place them in the binding machine aligning with the edge guide. [Move]
 - 2.2. Punch the short edge of the covers by pressing the handle bar down. Remove the covers from the machine. [Move]
 - 2.3. Extend the edge guide all the way out. [Discrete Event]
 - 2.4. Place the covers in the binding machine again, aligning them with the new edge guide placement. [Move]
 - 2.5. To secure the covers in place, push the alignment peg down through the punched hole underneath it. The alignment peg is the white plastic knob underneath the handle. [Discrete Event]
 - 2.6. Pull out pink peg number 5. [Discrete Event]
 - 2.7. Press the handle bar again to punch the remaining holes. [Discrete Event]
 - 2.8. Pull up the alignment peg to release the covers. Remove the covers from the machine. [Move]
 - 2.9. Push the edge guide all the way back in. [Discrete Event]
 - 2.10. Push the pink peg number 5 back in. [Discrete Event]
3. Punch the notebook pages
 - 3.1. Take 4 blank pages and place them in the binding machine aligning with the edge guide. [Move]
 - 3.2. Punch the pages by pressing the handle bar down. Remove the pages from the machine. [Move]
 - 3.3. Extend the edge guide all the way out. [Discrete Event]
 - 3.4. Place the pages in the binding machine again, aligning them with the new edge guide placement. [Move]
 - 3.5. To secure the pages in place, push the alignment peg down through the punched hole underneath it. [Discrete Event]
 - 3.6. Pull out pink peg number 5. [Discrete Event]
 - 3.7. Press the handle bar again to punch the remaining holes. [Discrete Event]
 - 3.8. Pull up the alignment peg to release the pages. Remove the pages from the machine. [Move]
 - 3.9. Push the edge guide all the way back in. [Discrete Event]
 - 3.10. Push the pink peg number 5 back in. [Discrete Event]
4. Prepare the notebook for binding
 - 4.1. Place the binding wire on the plastic hooks on the edge of the machine. The binding wire has wide loops on one side and narrow loops on the other. Place the wide loops on the plastic hooks, with every other wide loop on a hook. Make sure the opening of the binding wire, or the empty part of the 'C' shape it forms is oriented up. [Move]
 - 4.2. Place the inside pages on the narrow loops of the binding wire. We will trim the excess wire later. [Move]
 - 4.3. Thread the front cover onto the narrow loops of the binding wire, on top of the pages, with the outer face facing up. [Move]
 - 4.4. Thread the back cover onto the narrow loops of the binding wire, this time with its outer face down, towards the front cover. [Move]
 - 4.5. Remove the binding wire with the threaded notebook from the machine. [Move]
 - 4.6. Use the wire cutters to trim the excess wire from the end of the notebook. [Discrete Event]
5. Bind the notebook
 - 5.1. Push down and turn the knob on top of the machine to adjust the binding size to 5/8 inches. [Discrete Event]

- 5.2. With the notebook still threaded, place the binding wire into the cinch press area of the machine, above the long black strip. Make sure to place the binding wire such that the open part, or the empty part of the 'C' shape it forms, is flush against the vertical body of the machine. [\[Move\]](#)
- 5.3. Hold the binding wire in place by holding on to the paper, and do not put your fingers inside the press. Then, push the handle bar to cinch a portion of the wires. [\[Discrete Event\]](#)
- 5.4. Shift the notebook placement so that the un-cinched portion of the wires is now in the press. Make sure your fingers are out of the press, and push the handle bar again to cinch the remaining wires. [\[Discrete Event\]](#)
- 5.5. Remove the notebook from the press and fold back the back cover. [\[Move\]](#)

Task 6: Install the wheels on a skateboard



1. Mount the truck
 - 1.1. Push four screws through the holes from the top of the board and then flip the board over so it's upside down on the table. [Move]
 - 1.2. Position the riser onto the screws on the bottom of the deck with the slightly longer edge positioned outwards. [Discrete Event]
 - 1.3. Position a truck over the riser and screws. The bushings should be facing inwards pointing to the middle of the skateboard. [Discrete Event]
 - 1.4. Add a nut to each screw and tighten them with your fingers. [Invisible State]
 - 1.5. Further tighten each screw. Use a screwdriver to hold each screw in place on the top of the deck while you use the skate tool to tighten the nut from the bottom. Do this for all four screws. [Invisible State]
2. Insert bearings into the wheels
 - 2.1. Tilt your board to the side and put one bearing on the axle of your truck. [Discrete Event]
 - 2.2. Push your wheel hard into the bearing. You should feel the bearing click into place. [Invisible State]
 - 2.3. Remove the wheel from the axle. [Move]
 - 2.4. Install a second bearing into the other side of the wheel using the same procedure. Afterwards, remove the wheel from the axle. [Move]
 - 2.5. Now tilt your board so you can use the other axle of your truck to install bearings on the second wheel. Put one bearing on that axle. [Discrete Event]
 - 2.6. Push the second wheel hard into the bearing until it clicks into place, and then remove the wheel from the axle. [Move]
 - 2.7. Install a second bearing into the other side of the wheel using the same procedure. Afterwards, remove the wheel from the axle. [Move]
3. Attach the first wheel to the truck
 - 3.1. Place a spacer on the axle. [Discrete Event]
 - 3.2. Place a wheel on the same axle over the spacer. The graphic on the wheel should face outside. [Discrete Event]
 - 3.3. Place another spacer on the axle, on the outside of the wheel. [Discrete Event]
 - 3.4. Add a nut to the axle over the second spacer and use your fingers to tighten the nut. [Invisible State]
 - 3.5. Use your skate tool to further tighten the nut, but make sure not to overtighten. At the end, the wheel should spin freely. [Invisible State]
4. Attach the second wheel to the truck
 - 4.1. Place a spacer on the second axle and place the second wheel on the axle over the spacer. [Discrete Event]
 - 4.2. Place a second spacer on the axle, on the outside of the wheel, and then add a nut to the axle over the second spacer. Use your fingers to tighten the nut. [Invisible State]
 - 4.3. Use your skate tool to further tighten the nut, but again, make sure not to overtighten. [Invisible State]
5. Make final adjustments
 - 5.1. Make sure the trucks are securely attached, and the nuts are tightened properly. [Check]
 - 5.2. Adjust the kingpin nut if necessary to achieve the desired tightness or looseness for turning. [Check]
 - 5.3. Spin each wheel to ensure they are spinning smoothly and there is no resistance or wobbling. [Invisible State]

Task 7: Make a Margarita mocktail



1. Prepare the margarita glass with salt
 - 1.1. Squeeze half a lime onto the sponge of the lime section of the glass rimmer. [Add Amount]
 - 1.2. Place salt in the salt section of the glass rimmer. Make sure it's evenly covering the surface. [Add Amount]
 - 1.3. Take a margarita glass, hold it upside down, and tap and press the rim onto the sponge imbued with lime juice. [Invisible State]
 - 1.4. Tap the glass rim into the salt container. [Invisible State]
2. Prepare the ingredients
 - 2.1. Use the citrus squeezer to squeeze the second lime half into a small cup. [Add Amount]
 - 2.2. Using the jigger, measure and pour 1 ounce of fresh lime juice into the larger tin of a Boston shaker. [Add Amount]
 - 2.3. Pour in 0.5 ounces of simple syrup. [Add Amount]
 - 2.4. Pour in 1.5 ounces of zero-proof tequila. [Add Amount]
 - 2.5. Add 6 or 7 ice cubes in the tin. [Add Amount]
3. Shake and pour the cocktail
 - 3.1. Cover the large tin with the smaller tin, and push the top (smaller) tin down firmly to ensure a seal. [Invisible State]
 - 3.2. Using both hands to catch the tins, make sure they stay pressed together and shake the drink 7 to ten seconds, until you feel that the metal has become quite cold. Make sure you hold them tight, as the shaking process might have them come undone and the drink might spill. [Invisible State]
 - 3.3. Open the top tin. If it does not come out easily, you can tap it from the side, or try to rotate it up. [Move]
 - 3.4. Place a hawthorne strainer over the opening of the large tin. [Move]
 - 3.5. While holding the strainer over the top of the tin, tilt the tin and strain the drink into the prepared glass. [Add Amount]

Task 8: Make a whiskey sour mocktail



1. Prepare the ingredients
 - 1.1. Using the jigger, measure and pour one ounce of egg white into the larger tin of a Boston shaker. [Add Amount]
 - 1.2. Using the citrus squeezer, extract the juice from a lemon half into a cup. [Add Amount]
 - 1.3. Pour 2 ounces of non-alcoholic Kentucky 74 bourbon into the tin with the egg white. [Add Amount]
 - 1.4. Pour 3/4 ounces of the freshly squeezed lemon juice into the same tin. [Add Amount]
 - 1.5. Pour half an ounce of simple syrup into the tin. [Add Amount]
2. Dry shake to froth the cocktail
 - 2.1. Take out the spring from the Hawthorne strainer and place it into the same tin. This will help beat the egg when we shake the cocktail. [Discrete Event]
 - 2.2. Cover the large tin of the Boston shaker with the smaller tin, and push the top (smaller) tin down firmly to ensure a seal. [Invisible State]
 - 2.3. Using both hands to catch the tins, make sure they stay pressed together and shake the drink for about 30 seconds. As you shake it the spring inside will froth the egg white. Make sure you hold them tight, as the shaking process might have them come undone and the drink might spill. [Invisible State]
 - 2.4. Open the top tin. If it does not come out easily, you can tap it from the side, or try to rotate it up. The drink inside should look frothy. [Move]
 - 2.5. Remove the spring from the tin and set it aside. [Move]
3. Shake the cocktail with ice to chill
 - 3.1. Add 6 or 7 ice cubes to the tin. [Add Amount]
 - 3.2. Cover again the tin with the smaller tin and push the top down firmly to ensure a seal. [Invisible State]
 - 3.3. Again, using both hands to make sure the tins stay pressed together, shake the drink for 7 to ten seconds, until the tins become cold to the touch. [Invisible State]
 - 3.4. Open again the top tin. If it doesn't open easily, tap it from the side, or try to rotate it up. [Move]
4. Strain the drink into the glass
 - 4.1. Place the Hawthorne strainer without the spring over the top of the tin. [Move]
 - 4.2. While holding the Hawthorne strainer over the top of the tin, tilt and pour the drink into a rocks glass. Ideally, there should be some foam at the top. [Add Amount]
5. Garnish
 - 5.1. Place 3 or 4 drops of the Angostura bitters over the foam at the top of the mocktail. [Add Amount]
 - 5.2. Using the metal toothpick, trace a line through the bitter drops on top of the foam. This will reshape the drops to look like little hearts. [Move]

A5 DEMOGRAPHICS QUESTIONNAIRE

Below is the demographics questionnaire that each participant in the SIGMACOLLAB data collection study completed at the end of the study.

1. What is your age? (Circle one)

18-25 26-35 36-45 46-55 56-65 66+ Prefer not to say

2. What is your gender?

Man

Woman

Non-binary / gender diverse

Self-described:

Prefer not to say

3. What is your occupation?

4. How familiar are you with mixed, augmented, and/or virtual reality technologies?

Not familiar at all 1 2 3 4 5 6 7 Very familiar

5. How often do you use a VR or MR headset?

Never 1 2 3 4 5 6 7 Often (daily)

6. Do you own a VR or MR headset? No Yes. What kind?