

ARGESTUREAID: A Voice-Based, Adaptive, and Context-Aware Conversational Assistant for Supporting Mid-Air Gesture Discovery and Execution

ANJALI KHURANA*, Simon Fraser University, Canada

AMY KARLSON, Reality Labs Research, Meta Inc., United States

CHRISTOPHER COLLINS, Reality Labs Research, Meta Inc., Canada

MENGJIE YU, Reality Labs Research, Meta Inc., United States

VINAY JAYARAM, Meta Reality Labs, United States

HRVOJE BENKO, Reality Labs Research, Meta Inc., United States

PARMIT K CHILANA, Simon Fraser University, Canada

End-users of immersive Augmented Reality (AR) applications struggle to discover and execute unfamiliar mid-air gestures. We introduce ARGESTUREAID, a novel proof-of-concept conversational assistant that provides adaptive, context-aware assistance tailored to users' tasks and gestural errors. By combining users' voice-based task descriptions with formalized representations of hand gesture data together with LLMs, ARGESTUREAID dynamically supports users through: (1) *feedforward*, directing users toward correct pose, position, and placement before initiating gestures, and (2) *corrective feedback*, offering targeted, adaptive instructions to address user errors during discovery and execution. Our initial feasibility study (N=15), which compared users' real-world interaction data against a manually labeled reference dataset, shows that ARGESTUREAID achieved an overall accuracy of 81.3%. Our follow-up user study (N=15) evaluated ARGESTUREAID against a BASELINE LLM-based voice assistant across unimanual and bimanual gestures. Results indicate that ARGESTUREAID achieved higher task completion rates and was rated more favorably for perceived discoverability and usefulness. These findings suggest that adaptive, in-context feedforward and corrective feedback, grounded in formal gesture representations and observed error patterns, may support users during mid-air gesture discovery and execution, and help them refine their understanding of required gestures. We discuss key considerations for designing future adaptive, context-aware AR systems in AR environments.

CCS Concepts: • **Human-centered computing** → **Empirical studies in HCI**.

Additional Key Words and Phrases: augmented reality; mixed reality; gesture discoverability; adaptive help; in-context help

ACM Reference Format:

Anjali Khurana, Amy Karlson, Christopher Collins, Mengjie Yu, Vinay Jayaram, Hrvoje Benko, and Parmit K Chilana. 2026. ARGESTUREAID: A Voice-Based, Adaptive, and Context-Aware Conversational Assistant for Supporting Mid-Air Gesture Discovery and Execution. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 10, 2, Article 48 (June 2026), 31 pages. <https://doi.org/10.1145/3810212>

*Corresponding author

Authors' addresses: [Anjali Khurana](#) (corresponding author), Simon Fraser University, Burnaby, BC, Canada, anjali_khurana@sfu.ca; [Amy Karlson](#), Reality Labs Research, Meta Inc., Redmond, Washington, United States, akkarlson@meta.com; [Christopher Collins](#), Reality Labs Research, Meta Inc., Toronto, Ontario, Canada, chriscollins@meta.com; [Mengjie Yu](#), Reality Labs Research, Meta Inc., Redmond, Washington, United States, annaymj@meta.com; [Vinay Jayaram](#), Meta Reality Labs, New York, New York, United States, vinayjayaram@meta.com; [Hrvoje Benko](#), Reality Labs Research, Meta Inc., Redmond, Washington, United States, benko@meta.com; [Parmit K Chilana](#), Simon Fraser University, Burnaby, BC, Canada, pchilana@cs.sfu.ca.



This work is licensed under a [Creative Commons Attribution 4.0 International License](https://creativecommons.org/licenses/by/4.0/).

© 2026 Copyright held by the owner/author(s).

ACM 2474-9567/2026/6-ART48

<https://doi.org/10.1145/3810212>

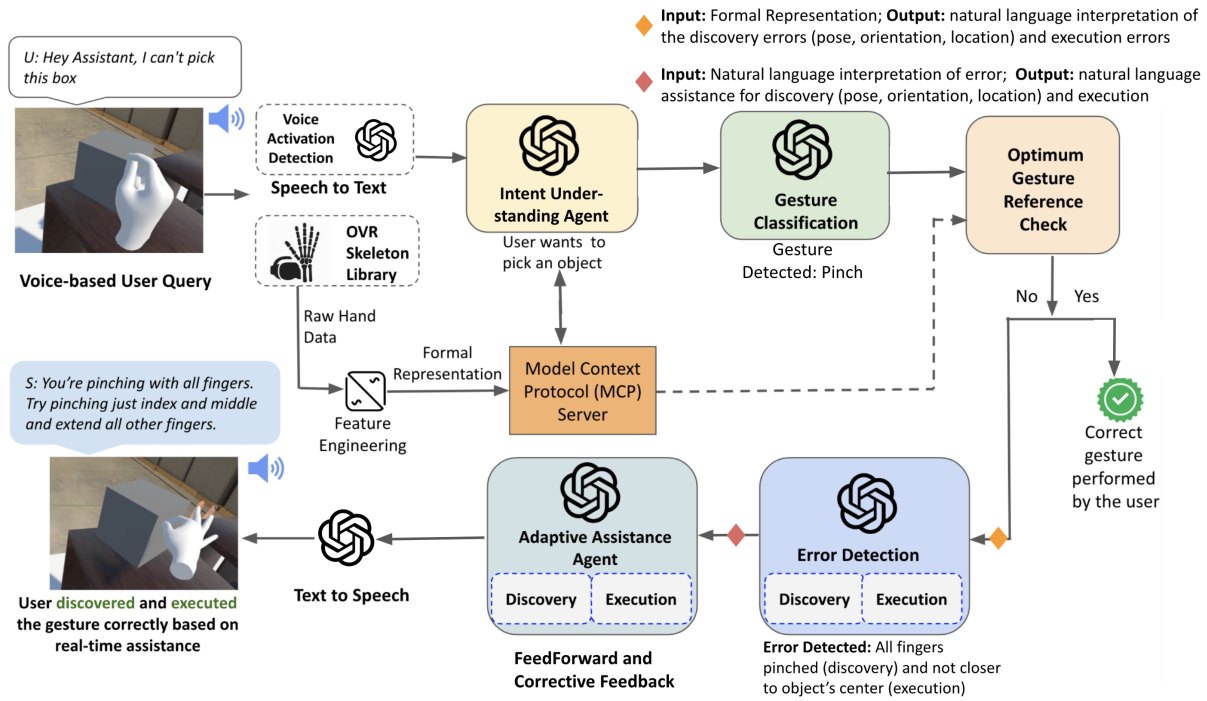


Fig. 1. ARGESTUREAid system architecture (version used in Study 2): User speech is interpreted to infer task intent and combined with structured gesture representations describing users' current hand pose and spatial relationships to virtual objects. This grounding allows the system to compare the intended and observed gesture states and to provide in-context feedforward and corrective feedback, delivered as spoken guidance during gesture discovery and execution.

1 INTRODUCTION

Mid-air gesture-based interactions in commercial AR headsets (e.g., Meta Quest, Microsoft HoloLens, Magic Leap) support tasks ranging from basic object manipulation to complex unimanual (e.g., grabbing, rotating objects) and bimanual (e.g., resizing virtual objects) gestures. However, novice users frequently encounter difficulties [45, 58] and struggle with forming accurate mental models of mid-air gestural interactions [35], affecting the adoption of many AR technologies. In particular, users struggle with *discovering* appropriate gestural inputs, and in *executing* the gestures with respect to virtual objects to perform tasks in immersive AR.

To support gesture discoverability and execution, existing onboarding solutions for mid-air gestural interaction focus on static tutorials [20], passive visual guides [20, 67] (e.g., dynamic 2D/3D gesture demonstrations as found in Meta's First Hand [49]), or multimodal audio-visual hints [25, 75] which users are expected to replicate and execute correctly for performing the tasks [20]. However, these methods provide generic, one-size-fits-all guidance suited mainly to simple hand gestures [20] and do not adapt to user-specific difficulties during gesture discovery or execution. Other solutions address user errors by improving gesture recognizer accuracy at the system level [58] or designing interaction techniques that proactively prevent errors [30, 68], or helping users to recover from errors [8, 12, 46]. Yet, such support is typically provided before or after the interaction, not during the critical moment of user failure. This challenge is amplified in complex tasks involving rich unimanual and bimanual gestures [35], which demand adaptive, error-aware, and context-sensitive guidance [35, 58] to assist users and support dynamic error recovery during gesture learning and execution, a gap our work addresses.

Addressing this gap requires overcoming two key challenges: 1) detecting users' gestural inputs and errors as they perform gestures, and 2) integrating gesture and error recognition data to provide context-aware, corrective guidance tailored to users' error state, task, and AR environment (e.g., virtual object). Recent advancements in large language models (LLMs)-based conversational assistance offer an avenue to revisit this challenge in providing context-sensitive, adaptive assistance that could dynamically support users in discovering and executing unfamiliar gestures. While LLMs are being increasingly explored in AR/VR contexts (e.g., [42]) and gesture-enhanced voice user interfaces (e.g., [71]), there has not been an explicit focus on support for adaptive error-aware guidance. A few studies have explored visual language models (VLMs) for gesture detection [73], but high latency and limited context-awareness restrict their suitability for adaptive, in-context assistance.

In this paper, we designed and implemented ARGESTUREAID, a novel proof-of-concept conversational assistant that provides adaptive, context-aware assistance tailored to users' tasks and errors during mid-air gesture discovery and execution in AR. ARGESTUREAID uses voice-based task descriptions and combines them with formalized representations of hand gesture data along with LLMs to interpret gesture states and detect user errors. By transforming gesture recognition data into an LLM-interpretable format, ARGESTUREAID dynamically supports users through two complementary assistance mechanisms during discovery and execution phases: (1) *feedforward guidance*, which proactively directs users toward the correct pose, position, and placement of gestures *before* initiation of each phase; and (2) *corrective feedback*, which provides targeted, adaptive instructions to address user errors during interaction. Our conversational approach delivers actionable, audio-based guidance tailored to user intent, detected error, and the surrounding AR context. The research questions guiding this investigation were:

- How can we design LLM-based adaptive, error-aware, and context-sensitive corrective guidance to support gesture discovery and execution in immersive AR?
- How do users perceive ARGESTUREAID's support for discoverability and execution across different tasks involving a range of unimanual and bimanual gestures, error types, and immersive AR contexts?

We conducted an initial feasibility study (N=15) with AR novices to validate ARGESTUREAID's accuracy by comparing users' real-world interaction data against a manually labeled reference dataset (ground truth). Our proof-of-concept ARGESTUREAID implementation achieved 81.3% overall accuracy across all AR tasks and users' gesture attempts, with higher performance during discovery at 85.1% (correcting pose and location errors) than during execution at 67.0% (correcting gesture placement relative to virtual objects). Insights from this study revealed ARGESTUREAID's challenges in latency, bimanual gesture support (See Table 1) and providing precise guidance relative to virtual objects during execution. To address these issues, we enhanced the data communication pipeline to reduce latency and expanded the formal gesture representation to improve execution-phase assistance for our subsequent user evaluation (See Figure 1).

To evaluate ARGESTUREAID from the user's perspective, we carried out a within-subject observational study (N=15) and compared ARGESTUREAID against BASELINE AR ASSISTANT, a state-of-the-art LLM-based voice assistant. Our findings indicate that ARGESTUREAID achieved higher task completion rates and was rated more favorably for perceived discoverability, and usefulness, especially for complex unimanual and bimanual tasks. Participants appreciated ARGESTUREAID's combination of *corrective feedback* which dynamically assisted them in refining pose, orientation and interaction with virtual objects during user errors, and *feedforward guidance* which helped identify the required gesture and reduce random trial-and-error behavior stemming from incorrect mental models, as seen in other studies [35]. In contrast to BASELINE AR ASSISTANT's generic and static guidance, participants emphasized that ARGESTUREAID's ongoing, gesture-aware feedback was essential for improving gestures during discovery and execution.

The key contributions of this paper are as follows: (1) a formal, structured hand-gesture representation interpretable by LLMs, which uses salient kinematic properties from hand-tracking data and enables the mapping

of continuous kinematic input to meaningful, situation-specific guidance during mid-air gesture discovery and execution; (2) a proof-of-concept system that operationalizes this approach to provide in-context, adaptive, and error-aware assistance during ongoing gesture discovery and execution by integrating users' voice-based task descriptions, structured hand-gesture representations, and LLM-based semantic mediation to generate feedforward and corrective feedback grounded in users' current gesture state and task intent during mid-air gesture discovery and execution; and (3) empirical insights from two user studies characterizing the value of conversational gesture assistance, indicating that ARGESTUREAID supports perceived discoverability and usefulness compared to a general-purpose conversational assistant without access to gesture state.

2 RELATED WORK

This research builds upon prior work on user interaction challenges and errors in discovering and using gestural interaction, context-aware solutions for gesture guidance, conversational and voice interfaces for guidance in AR.

2.1 User Interaction Issues and Errors When Using Gestural Interactions in AR

AR applications enable users to use often unfamiliar mid-air hand gestures to interact with virtual 3D objects integrated into their immediate physical surroundings [15, 29, 35]. In discovering and performing these gestures, users commonly encounter two types of errors—**system errors** and **user errors**. System errors involve failures that occur when the AR system fails to recognize correct gestures (false negatives) or when the system mistakes incorrect gesture input to be correct (false positives) [35, 45, 58]. Both of these system errors degrade user experience but can often be mitigated by refining recognizer algorithms or clarifying user input [45, 58, 59]. In contrast, user errors arise from users' misconceptions, such as transferring real-world object interactions to virtual objects in AR or making incorrect assumptions about positions, orientations, pressures, or angles needed for proper execution [26, 31, 35, 52, 66, 72]. These errors reflect challenges in users' mental models, discovering and accurately executing unfamiliar AR gestures, and are more difficult to recognize and address because they stem from behavior and cognition rather than system failures. Prior work highlights that user errors dominate user interaction with immersive AR compared to system errors [35] and that correcting these errors often requires techniques beyond optimizing gesture recognition. However, much of the existing research focuses on preventing and mediating system errors, leaving support for user errors relatively underexplored in practice [45, 58, 59].

Some studies have taken initiative to mitigate the impact of user errors by designing interfaces that facilitate reversibility (e.g., history mechanisms or undo buttons) [8]. Other solutions have focused on designing systems to detect the occurrence of user errors and using those instances to refine and personalize the recognizer models [58]. However, these solutions alone do not help users develop accurate mental models of gestural interactions nor guide them in discovering and executing unfamiliar gesture-based tasks. An ideal solution to address user errors in AR interactions would involve ongoing detection of these errors, followed by adaptive error recovery techniques that guide users toward successful gesture discovery and execution [35, 58]. Although adaptive assistance is promising, prior ML-based systems support only limited errors types and lack contextual understanding. In AR-based gestural interaction, a single gesture error (e.g., an incorrect grasp) can arise from many different low-level issues, such as incorrect finger position, hand orientation, or movement timing. Unlike traditional AI systems based on simpler models [23, 28, 51, 56], which are limited to handling a narrow range of application-specific queries, generative AI foundation models are capable of tackling complex, open-ended tasks. Recent LLM advances enabling interpretation of diverse user context and generation of natural language feedback [37, 42, 71] open new possibilities for revisiting this challenge. Recent studies have explored the use of VLMs for gesture detection [73]. While promising in their ability to process multimodal inputs, these approaches are often computationally intensive, incur high processing latency [43], and struggle to incorporate nuanced context of user errors [73]. Additionally, VLMs are prone to misclassification when interpreting subtle variations [43]

in gesture input arising due to user errors, making them less suitable for ongoing, interactive assistance in AR environments. Our design and implementation of ARGESTUREAID introduces a novel LLM-based approach that integrates users' voice-based task descriptions with formalized representations of hand gesture data to detect ongoing mid-air gestures and identify user errors, enabling adaptive and context-aware guidance as users perform their gestures.

2.2 Context-Aware Gesture Guidance and Onboarding Solutions

Prior research on user onboarding and gesture learning in AR interfaces spans a diverse range of techniques, beginning with static step-by-step training or tutorials [20] that guide novices through simple gestures during an initial onboarding phase to displaying a static icon signifying the gesture or a dynamic 2D/3D demonstration of the gesture, such as in Meta's First Hand [49], audio-visual hints [75], gaze-based assistance, and audio-visual stimuli [25]. However, these approaches focus on general onboarding experiences rather than providing targeted, adaptive guidance that mitigates user-specific errors or misconceptions during the discovery and execution of complex gestures [20, 67]. In an effort to support continuous learning, prior research has explored *feedforward* and *feedback* approaches, often separately. For example, OctoPocus [13] and ShapelineGuide [10] use dynamic visual cues to preview correct gesture forms and provide immediate performance feedback. In contrast, other work [50] discusses feedforward strategies for revealing possible gestures before an action is performed. Our ARGESTUREAID approach uses the benefits of both feedforward to cue users before each phase of gesture discovery and execution and corrective feedback to assist with recovery of user errors during each phase.

Although these approaches have proven useful for 2D gestural interactions on interactive displays, they do not directly transfer to mid-air gestures in immersive AR, which involve more complex 3D object manipulations and nuanced pose and gesture execution requirements relative to 3D virtual objects within real-world environments [27]. Specific to the AR space, prior literature has explored micro visual guidance [67] that helped users to precisely perform single-handed and double-handed static pose gestures. While such help approaches assist users to provide micro-level help for basic static pose gestures interactions, they often do not adapt to evolving user needs or complex tasks, thereby limiting their ability to address ongoing issues and dynamic user errors [35, 45, 58] that are prevalent as the complexity of gestures increases. Moving beyond simpler hand poses, complex AR/VR gestures are pivotal in advanced scenarios such as expressive social communication (e.g., VRChat [64]), skills training requiring precise gesture learning [60], productivity-focused tasks [1], and rich gaming interactions [17]. Building on these works, we implemented and evaluated ARGESTUREAID across diverse unimanual and bimanual gestures, including static pose-and-path, dynamic pose, and dynamic pose-and-path, through a two-part study: a feasibility study assessing accuracy, followed by an observational user study offering insights into users' perceptions of gesture discoverability and usefulness of ARGESTUREAID compared to BASELINE AR ASSISTANT in immersive AR.

2.3 Conversational Interaction with Voice Interfaces

Conversational voice interfaces have long been appreciated for their natural interaction style and expressive power; however, they often face design challenges that limit their use to brief and low-risk tasks (e.g., reminders or simple task interactions) [33, 47, 54]. Prior work underscores the importance of considering user context when designing voice commands [71], especially in AR environments where users must manage both real-world and virtual objects [14, 35]. Recent studies further highlight the need to provide minimalistic, precise instructions [33, 54] to prevent overwhelming users, particularly in situations where gesture discovery and execution are prone to errors. Consequently, well-structured and succinct assistance is vital for preventing overload while ensuring effective, context-aware guidance. In ARGESTUREAID, we focus on two critical aspects of structuring

help: (1) pose and location (gesture discovery) and (2) gesture placement relative to the virtual object (gesture execution), as users attempt the majority of user errors during discovery and execution [35].

Prior work has explored the variety of multimodal assistance strategies (e.g., using text [16], visuals [39, 62], haptics [55], gaze [48, 74], audio [21] or their combination [42]). In particular, research on voice-based interactions has integrated speech with gaze [42], touch [41, 65], and additional input modalities [55] to provide richer contextual cues for tasks such as object retrieval or foreign language translation in AR [42]. However, some studies have raised concerns that overloading users with multiple, and unfamiliar modalities (e.g., excessive visual cues [40]) can lead to sensory or cognitive clutter, thus complicating rather than clarifying gesture-based interactions. These issues are especially problematic in gesture-driven AR systems where discoverability and correct execution remain significant challenges [35]. Recent work has demonstrated that conversational and audio-based interactions can be effective for gesture support [24, 53]. In this work, we focus on a voice-based conversational approach that can adaptively tailor instructions to each user's needs without cluttering the AR interface, especially for novice AR users. We explored this by designing and implementing two interventions: ARGESTUREAID and BASELINE AR ASSISTANT, described below.

3 DESIGN AND IMPLEMENTATION OF ARGESTUREAID

We designed and implemented ARGESTUREAID, an adaptive, context-aware conversational assistant for mid-air gestures that integrates formalized hand gesture data with LLM reasoning. The system runs on Meta Quest 3 using OVR Skeleton tracking [6], GPT-4o based LLM agent, and Unity's mixed reality toolkit. Figure 1 shows the overall architecture of ARGESTUREAID, which can be adapted to any AR headset supporting compatible hand gesture data.

3.1 Structuring the Assistance for Gesture Discovery and Execution

Recent studies [35, 58] have shown that while system-level recognition errors in immersive AR are significant, user errors, primarily stemming from incorrect mental models, dominate gesture discoverability and execution issues in user interaction. During gesture discovery, users frequently struggle to discover the correct pose, including precise position, orientation, and spatial location [35], required to initiate a range of gestures, from static pose and path to dynamic ones. For example, in a swiping menu task, a static pose and path gesture involves maintaining a fixed pose of extended index and middle fingers (position, orientation, and specific hand configurations) while moving the hand through space (location) in different directions. Furthermore, during the gesture execution, users often faced issues due to confusing gesture-to-task mappings and misaligned assumptions about correct placements of the gestures [35] (e.g., distance between gesture and virtual object) in relation to virtual objects, which hindered successful task completion.

To support gesture discovery and execution, ARGESTUREAID should provide guidance on key hand features, including pose and placement relative to virtual objects during gesture execution. To enhance the assistance, *feedforward* [20, 35, 63] providing proactive indications about what gestures users are expected to do even before doing tasks has shown to be effective in aiding discoverability. Such guidance can help mitigate discoverability issues stemming from incorrect mental models, as it clarifies user expectations before they initiate a gesture. For effective error recovery, *corrective feedback* that goes beyond binary success-or-failure cues is helpful [20]. For example, offering in-context, adaptive feedback that analyze users' gestures and error states to inform users *where* and *why* they failed, and suggest *how to improve* can assist users in gesture discovery and execution. Such ongoing, in-context assistance can steer users toward correct interactions when they begin to go down the wrong path.

Informed by these insights, we developed four key design goals for ARGESTUREAID to support ongoing gesture discovery and execution across a range of unimanual and bimanual gestures.

- **DG1 Structuring the assistance to guide hand features and gesture-to-virtual object location:** Provide assistance on guiding critical hand features—pose (position, orientation) and spatial location for gesture discovery and gesture’s placement relative to the virtual object for gesture execution.
- **DG2 Feedforward for Gesture Discovery and Execution:** Begin by providing proactive (feedforward) guidance on the correct pose along with spatial location needed for the gesture discovery and correct gesture placement relative to the virtual object for gesture execution of the intended task, preparing users before they begin the required interaction.
- **DG3 Corrective Feedback for Gesture Discovery and Execution:** When discovery and execution errors occur, the assistant should provide corrective feedback tailored to the user’s current error state, specifying *where* and *why* error occurred, and suggesting *how to improve* their pose and location (for discovery) and gesture’s placement relative to the virtual object (for execution).
- **DG4 Context-aware, Adaptive Conversational Assistance:** Provide context-aware and conversational feedforward and corrective feedback (audio-based in our case) for gesture discovery and execution, based on ongoing detection of user gesture and error states as users perform their gestures.

3.2 Formal Gesture Representation

A significant challenge in providing adaptive assistance for gesture discovery and execution involves creating robust, interpretable representations for hand pose, motion trajectories, and interactions with virtual objects. While VLMs have been utilized for visual processing, they face limitations in computational cost, latency, and reliability [43]. Using hand-bone joints coordinates data from hand tracking libraries, such as the OVR Skeleton [6], which relies on multi-angle camera imagery and computer vision algorithms, is more practical; however, raw sensor data is unstructured, numerical and not directly interpretable by LLMs. To address this, we developed a structured, formal representation (Figure 2) that converts raw skeletal data into compact, LLM-comprehensible inputs, supporting two key phases:

- **Gesture Discovery:** Identifying hand pose (position, orientation) and dynamic trajectories over time (location).
- **Gesture Execution:** Determining gesture placement relative to virtual objects.

3.2.1 Formal Gesture Representation for Gesture Discovery. We iteratively refined structured gesture representations to support accurate pose, location, and execution reasoning. Early experiments using raw coordinates and temporal differencing alone in the hand skeletal data were insufficient for capturing fine-grained pose and dynamic gesture trajectories. Our final gesture representation therefore encodes higher-level, interpretable features: pairwise fingertip distances, finger joint angles, hand centroid trajectories, and wrist rotations. This enables robust detection across static, dynamic, and bimanual gestures (Figure 2). Details of the iterative experiments are provided in Appendix A.

Proposed Higher-Level Features and Full Formal Representation To enhance pose and location detection accuracy, we preprocessed the raw hand coordinates to extract high-level features. Features representing pose include:

- **Pairwise Fingertip Distances:** We compute the 3D Euclidean distance for each pair of fingertips (e.g., thumb-to-index, thumb-to-middle), producing a compact feature set that reliably discriminates pinches, grips, and extended-finger poses.
- **Finger Angles:** For each pair of adjacent fingers (e.g., thumb-to-index, index-to-middle), we compute the angles between their direction vectors. This compact feature captures finger spread, distinguishing gestures such as open-hand poses or closed-hand postures; Furthermore, features representing Location include:

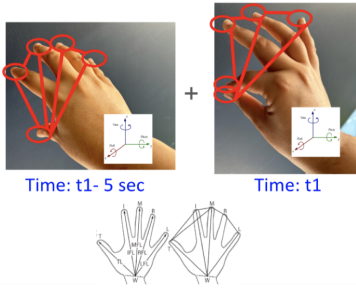
Approach: Using preprocessed hand data and sending Δ (current data t1 and data before 5 seconds t1-5) to LLM		
	Phases	Formal Representation of hand gesture data used [Input to the LLM]
	Discovery	Pose: fingertip distances [<i>distance_thumb_index</i> , <i>distance_index_middle</i> , <i>distance_middle_ring</i> , <i>distance_ring_pinky</i> , <i>distance_thumb_middle</i> , <i>distance_thumb_ring</i> , <i>distance_thumb_pinky</i>], finger joint angles [<i>theta_thumb</i> , <i>theta_index</i> , <i>theta_middle</i> , <i>theta_ring</i> , <i>theta_pink</i>] (for detecting curls, extended fingers) Orientation: Wrist Rotation angles [<i>pitch</i> , <i>yaw</i> , <i>roll</i>]- *only for Dynamic pose (Rotation) Location: Hand centroid trajectory [<i>Centroid</i> (<i>x1</i> , <i>y1</i> , <i>z1</i>)]
	Execution	Gesture Placement: Object Center Distance [<i>distance_handcentroid_objectcenter</i>]

Fig. 2. Formal representations of hand gesture data used in ARGESTUREAID for both unimanual and bimanual gestures across two key phases: Gesture Discovery and Execution. These representations are integrated into our error detection framework, enabling accurate identification and differentiation of a wide range of gestures. *Since Dynamic pose gesture for rotating a box (highlighted), requires additional orientation data, so we capture rotational motions through wrist rotations.

- **Temporal Dynamics:** We retain the time-differential approach (hand data from current frame vs. snapshot 5 seconds prior), capturing velocity and direction for relevant joints.
- **Hand Centroid and Wrist Rotation:** We compute the hand centroid by averaging all five fingertip coordinates to capture overall hand position. Time-differential processing of the centroid encodes linear hand motion for unimanual gestures (e.g., swiping, zooming), while changes in distance between both hand centroids capture cooperative interactions during bimanual gestures. Wrist rotation angles (pitch, yaw, roll), processed temporally, further encode rotational gestures such as twists and rotations.

3.2.2 Formal representation for Gesture Execution. To determine proximity and assist users during execution, we considered **Object Center Distance**, distance from the hand centroid to the virtual object. For example, if a user needs to pick up a virtual object, the LLM can guide them using this distance measure.

Together, these formal representations for discovery (Section 3.2.1) and execution (Section 3.2.2) are integrated into our error detection framework (See Figure 1). They serve as LLM inputs, enabling error detection and recognition of diverse gestures, making them generalizable across these range of gestures (See Figure 2).

3.3 Proof of Concept ARGESTUREAID Implementation: Adaptive Error Detection and Assistance Module

Users can initiate the ARGESTUREAID by saying “Hey Assistant” and making a verbal query. ARGESTUREAID then provides context-aware guidance through a multi-stage pipeline integrating user speech input, gesture classification, adaptive error detection, assistance framework, and TTS-based feedback.

3.3.1 Intent Understanding and Gesture Classification. ARGESTUREAID transcribes the verbal query via Open AI Whisper and maps it to high-level AR tasks (e.g., object pickup) through a few-shot intent understanding prompt. The identified intent is subsequently matched against a curated gesture set via a classification prompt, selecting the most appropriate gesture (e.g., “pinch-to-rotate”).

Study 1 (Feasibility Study): Tasks

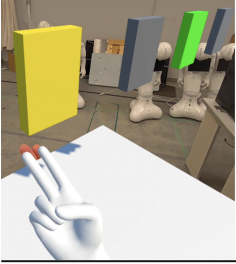
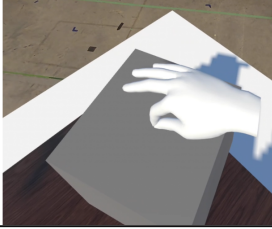
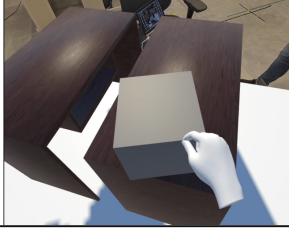
Unimanual Gestures [one hand]			Bimanual Gestures [two hands]
Swiping Tiles	Rotating a box	Grabbing a box	Zooming a rectangle
			
Static Pose and Path	Dynamic Pose	Dynamic Pose and Path	Dynamic Pose and Path
<div style="text-align: center;">  Gesture Form [Increasing task complexity] </div>			

Fig. 3. Overview of Tasks and Gestures explored in Feasibility Study: (a) Static Pose-and-Path (“Swiping tiles”): users swipe with two fingers to move from the current tile (yellow) to the target tile (green). (b) Dynamic Pose (“Rotating a box”): users rotate a box clockwise to adjust its orientation upside down. (c) Dynamic Pose-and-Path (“Grabbing a box”): users grab a box from one table and move it to the center of the second table. (d) Bimanual Dynamic Pose-and-Path (“Zooming a rectangle”): users perform a zoom gesture to enlarge a rectangle from both sides.

3.3.2 Error Detection. Next, as shown in Figure 1, the LLM agent converts formal gesture representation data into a natural-language description, identifying pose errors and spatial alignment relative to virtual objects (e.g., “thumb and index finger are pinched” during discovery; “hand is far from the object” during execution error). This natural language description allows for a precise evaluation of how closely the user’s gesture aligns with the optimal target gesture (See Table 5) to provide ongoing, in-context assistance. We isolate error detection into its own prompt to improve accuracy, as combining detection and assistance generation increased complexity and reduced performance in early testing.

3.3.3 Adaptive Assistance (for Gesture Discovery and Execution). We input the user’s current textual description of pose, orientation, location, and gesture placement relative to virtual object, along with the “ideal discovery and execution” for the intended gesture (expressed in natural language) and the formal gesture representation into an Adaptive Assistance LLM Agent, enhanced with chain-of-thought [69] instructions (See Figure 1). Based on the information provided, it generates context-relevant corrective instructions in two phases:

- **Gesture Discovery (Pose and Location Correction):** Once the user’s intended task and gesture are identified, the system begins by providing feedforward on the correct pose and location. As the user performs the gesture, the ERROR DETECTION module first identifies ongoing pose errors, relating to the position and orientation of the hand and fingers, and provides corrective feedback accordingly. Following the pose adjustment, ARGESTUREAID addresses location issues by offering corrective feedback to properly align the hand’s dynamic motion based on the identified user’s current gesture and error state.
- **Gesture Execution:** Once the user has achieved the correct pose and location, the system proceeds to provide instructions for execution, specifically tailored to interact accurately with virtual objects.

Finally, corrective instructions are delivered via speech synthesis to Quest 3 every 5 seconds, providing iterative feedback in response to user queries or gestures until the gesture is successfully executed or the user’s intent changes.

4 FEASIBILITY STUDY: VALIDATING GESTURE-ERROR DETECTION AND GUIDANCE ACCURACY

We conducted a feasibility study to assess whether ARGESTUREAID could detect gesture states and errors with sufficient accuracy to support feedforward and corrective assistance. The goal of this study was not to evaluate user experience or performance outcomes, but to validate the technical viability of our proof-of-concept system prior to a comparative user evaluation (Section 5). Fifteen participants (8F/7M), all new to mid-air gesture interaction) completed a set of representative unimanual and bimanual AR tasks using a Meta Quest 3 headset (Fig. 3).

4.1 Study Design and Analysis

Choice of Gesture Forms Across Two Studies: We drew inspiration from recent literature (e.g., Hosseini’s survey [32]) which classifies mid-air gestures supported by commercial AR systems into four categories (See Figure 3): static pose (fixed hand configuration, e.g., pointing or tapping), static pose with path (fixed hand pose with spatial movement, e.g., swiping), dynamic pose (changing hand configuration without spatial movement, e.g., pinch-rotate), and dynamic pose with path (changes in both pose and position, e.g., grabbing or zooming objects). Our tasks across both the studies were designed to cover this full spectrum of complexity, from static pose and path to dynamic pose and path gestures, for both unimanual and bimanual interactions in realistic virtual-object scenarios.

Each session began with a brief introduction to AR and general interaction tips (e.g., using hand gestures). Participants then completed four tasks (2 unimanual and 2 bimanual) using an AR application on the Meta Quest 3 headset. Tasks were selected to cover common gesture primitives and error-prone interaction patterns, including hand pose configuration, orientation, and spatial positioning relative to virtual objects (See Figure 3). Participants interacted with the AR application using mid-air hand gestures, while ARGESTUREAID monitored gesture execution and provided assistance as designed. Interaction data, system logs, and screen recordings were collected for analysis. There was at least one researcher present in the room to observe each session and take notes.

Participants could optionally access device-level help resources (e.g., [49]), but were not required to complete a tutorial. Consistent with prior findings that users often skip tutorials [19, 35], participants were instead encouraged to rely on ARGESTUREAID for assistance during gesture discovery and execution. Each task was presented sequentially, with up to 8 minutes allotted per task. Participants were encouraged to think aloud, and were reminded that the study focused on how they interacted with ARGESTUREAID, rather than on task performance or proficiency with AR.

To support evaluation, we collected a corpus of interaction data, including participants’ help-seeking utterances, gesture execution traces, system logs of detected gesture states, and corresponding LLM-generated assistance. Video recordings were used to identify ground-truth gesture states and errors. This corpus enabled two analyses: (1) error detection accuracy, assessed by comparing system-detected error states against manually labeled gesture errors derived from video recordings (Section B.3.1); and (2) assistance accuracy, assessed by comparing the system’s generated assistance against ideal responses grounded in the formal gesture representation (Section B.3.2).

We used an expert-rating approach: the experimenter, in consultation with the research team, compared system outputs against a manually labeled reference dataset representing ideal assistance (Section B.3). Participants’ think-aloud commentary was used to contextualize system behavior and to identify opportunities for improving assistance strategies.

4.2 Results

4.2.1 Accuracy of ARGESTUREAID’s Adaptive Assistance in Gesture Discovery and Execution. To support users during gesture discovery and execution, ARGESTUREAID achieved an overall assistance accuracy of 81.3% across

Table 1. (i) **Error Detection Accuracy:** ARGESTUREAID demonstrated the highest accuracy in detecting errors among unimanual gestures, including dynamic pose and path (grabbing), followed by dynamic (rotation) and static pose-and-path gestures. These scores decreased for bimanual gestures with dynamic location, such as dynamic pose-and-path two-hand zoom. (ii) **Accuracy in Providing Adaptive Assistance:** ARGESTUREAID's assistance achieved higher accuracy during gesture discovery than execution across all gestures. Among different gestures, ARGESTUREAID demonstrated higher accuracy for complex unimanual (highlighted in grey) compared to bimanual gestures.

Gesture Form	Error Detection Accuracy	ARGESTUREAID's Adaptive Assistance Accuracy		
		Overall	Discovery	Execution
Dynamic Pose-and-Path [Grabbing a box]	 85.8%	 91.1%	 94.3%	 78.0%
Dynamic Pose [Rotating a box]	 83.7%	 87.9%	 92.4%	 71.5%
Static Pose-and-Path [Swiping tiles]	 72.9%	 76.8%	 79.5%	 61.4%
Dynamic Pose-and-Path [Zooming a rectangle]	 67.1%	 69.2%	 74.1%	 57.1%

all AR tasks and participants' gesture attempts, demonstrating that it could reliably detect the gesture states and errors necessary for the user-facing evaluation in Study 2. ARGESTUREAID achieved high accuracy (85.1%) in assisting with pose and location error correction during gesture discovery, but struggled to identify errors related to proximity to virtual objects (e.g., how close the user should approach) and to pinpoint the precise interaction points required for gesture execution (e.g., border vs. center of a virtual box). Consequently, ARGESTUREAID's accuracy during gesture execution, specifically for correcting gesture placement relative to virtual objects, was lower at 67.0% across all gesture types.

Across different gestures, ARGESTUREAID demonstrated higher overall accuracy across complex unimanual gestures compared to bimanual gestures (see Table 1). Lower error detection accuracy scores (67.1%) for bimanual gestures stemmed from limitations in the current formal representations, which could not accurately detect simultaneous pose errors (e.g., two-hand pinch gesture) across both hands for 7/15 participants during gesture discovery, particularly when participants' fingers were positioned closely together, which in turn led to repeated, incorrect feedback affecting the accuracy of ARGESTUREAID's assistance. Similar inaccuracies occurred for 6/15 users performing static pose-and-path gestures, where extending the index and middle fingers for tile swiping was incorrectly flagged as also including the ring and pinky fingers.

Consistent with these trends, assistance accuracy across discovery and execution phases was highest for dynamic pose-and-path gestures, followed by dynamic pose and static pose-and-path gestures, while bimanual dynamic pose-and-path gestures demonstrated comparatively lower accuracy (See Table 1 for ARGESTUREAID's assistance accuracy).

4.2.2 Areas of Improvement in ARGESTUREAID. Despite ARGESTUREAID's high accuracy for complex dynamic gestures, average task completion times remained longer (dynamic pose-and-path: 2 min 30 s; dynamic pose: 1 min 30 s; static pose-and-path: 1 min). This delay stemmed from limitations in the current formal representation, particularly during gesture execution, where users struggled to gauge how and where to interact with virtual

Table 2. Comparison of two AR Assistants: BASELINE AR ASSISTANT (our implementation inspired from state-of-the-art LLM Powered Voice Assistants) vs. ARGESTUREAID (our novel in-Context Adaptive Assistant Grounded in Gestural Formal Representation).

	ARGESTUREAID	BASELINE AR ASSISTANT
In-context, ongoing Assistance	Provides ongoing assistance, in-context tailored to user queries, gestures and error states during gesture discovery and execution. Voice-enabled assistance.	Provides ongoing assistance in-context tailored to users' queries Voice-enabled assistance.
Formal Gesture Representation	Integrated formal representations of hand gesture data with LLMs to interpret gesture states and user errors.	No
Step-by-Step Assistance	Provides step-by-step guidance during gesture discovery (pose, location and orientation) and execution (gesture placement relative to the virtual object). Provides combination of <i>feedforward</i> to proactively guide users toward the correct gesture before each phase, and <i>corrective feedback</i> to address errors during interaction.	No
Follow-up questions	Enables follow-up questions using the latest context from user's query, as well as gesture, and error state.	Enables follow-up questions using the latest context from user's query.

objects. Feedback latency also contributed: input was sampled at fixed 5-second intervals, causing delays between gesture attempts and corrective responses during both discovery and execution.

5 STUDY 2: USER EVALUATION

To assess task completion, perceived discoverability, and usefulness, we carried out a within-subject observational study with follow-up interviews, comparing ARGESTUREAID against BASELINE AR ASSISTANT, our implementation of a state-of-the-art LLM-based voice assistant.

5.1 Improvements in Intervention 1: ARGESTUREAID

Based on insights from our feasibility study, we implemented three key enhancements to ARGESTUREAID (See Figure 1). **(1) Enhanced Formal Representations:** We introduced an *Object Edge Distance* parameter, distance from the hand centroid to each object edge, to better capture bimanual coordination, simultaneous pose errors, and spatial proximity (e.g., distinguishing interactions at the border vs. center of virtual objects), improving ARGESTUREAID's precision in providing feedback during gesture execution; **(2) Adaptive Recognition via Model Context Protocol (MCP):** To address latency issues and improve scalability, we integrated the MCP [11] into ARGESTUREAID's architecture, enabling adaptive, event-driven gesture detection instead of fixed-interval sampling. The MCP server allows the Intent Understanding Agent to intelligently request gesture input only when needed for error detection, reducing computational overhead and enhancing system responsiveness; **(3)**

Voice Activity Detection and Experimenting with LLMs: We incorporated *Voice Activity Detection (VAD)* [5] to capture voice input only when speech is detected, minimizing continuous polling. We also experimented with LLMs and used GPT-4o-mini for transcription and GPT 4.1 for error detection to enable low-latency interaction.

With these improvements, ARGESTUREAID provides feedback both in response to user queries and/or to performed gestures, reducing gesture detection latency to 13ms, with the overall end-to-end latency of 25–30ms (from gesture detection to receiving audio output). Details of the prompts sent to the LLM for each module of ARGESTUREAID are provided in the Appendix C.

5.2 Intervention 2: Design and Implementation of Baseline AR assistant

Our second intervention for the follow-up evaluation was inspired by state-of-the-art LLM-powered voice assistants, such as ChatGPT Voice, Bing Chat with GPT-4 [3, 61]. These assistants support flexible, context-aware interactions, including multi-turn dialogue, natural language understanding, interactive clarification, and diverse task execution, moving beyond rigid command-based systems to more adaptive, human-like handling of general voice queries. Recent efforts have explored integrating such assistants into AR headsets and wearables (e.g., Meta AI Voice [4]); however, these implementations are typically standalone tools and do not function as in-application assistants within AR environments. To establish a practical and controlled comparison point for the user study, we implemented our own Baseline AR assistant directly within the Meta Quest 3 headset using GPT 4.1 (same model as ARGESTUREAID). This setup enables consistent data collection in real use-case scenarios and allows systematic evaluation of the proposed ARGESTUREAID framework against a widely recognized voice-based assistant.

5.3 Study Design

5.3.1 Participants. We recruited participants (7F|7M|1 Non-binary) who had little to no prior experience with AR or VR from our university’s mailing lists and snowball sampling. Our participants came from both CS (9/15) and non-CS (6/15) backgrounds (Criminology, Business, Statistics, Information Visualization, Sociology) and included information designers, students, researchers, and software developers. They represented a range of age groups: 18-24 (50%), 25-34 (44%), 35-44 (6%) and had different levels of education (5 Bachelor’s, 4 Master’s, 7 PhD). All participants in Study 2 were new and had not participated in Study 1.

5.3.2 Choice of Tasks and Gestures. Similar to Study 1, we drew on recent literature (e.g., Hosseini’s survey [32], [67]) to design a total of six tasks for Study 2, covering the full spectrum of gesture complexity, from static pose-and-path to dynamic-pose-and-path, across both unimanual and bimanual interactions (See Section 4.1 for details). We extended ARGESTUREAID’s implementation on Meta Quest 3 to include 6 mutually exclusive and non-overlapping gestures to minimize cross-condition learnability. Each gesture within the task was chosen to ensure distinct hand poses, affordances, and execution relative to the virtual object. For Study 2 (Figure 4, Table 6), we retained the Swiping Tiles (Static Pose-and-Path), Rotating a Box (Dynamic Pose), and Zooming a Rectangle (Bimanual Dynamic Pose-and-Path) tasks from Study 1 (shown in Figure 3) and introduced new tasks to expand the evaluation: Grabbing Sphere (Static Pose-and-Path), Increasing Box Size (Dynamic Pose), and Create a Sphere in 3D (Bimanual Dynamic Pose-and-Path). The new tasks involved gestures distinct from the previous ones, such as two-handed circular motions or thumbs-up with lateral movements, designed to minimize cross-condition learnability.

5.3.3 Procedure. We used a within-subject design to reduce participant variability, employing a Latin Square counterbalancing [57] method to randomize the order of the two AR assistants conditions. The same apparatus and in-lab setup from the first study were used to conduct this second study. Each session started with a brief introduction to the concept of AR, along with general tips for interacting with the application (e.g., using hand gestures). Participants then perform 3 AR tasks per AR assistant condition, referring to each gesture form (Static

Study 2 (User Evaluation): Tasks

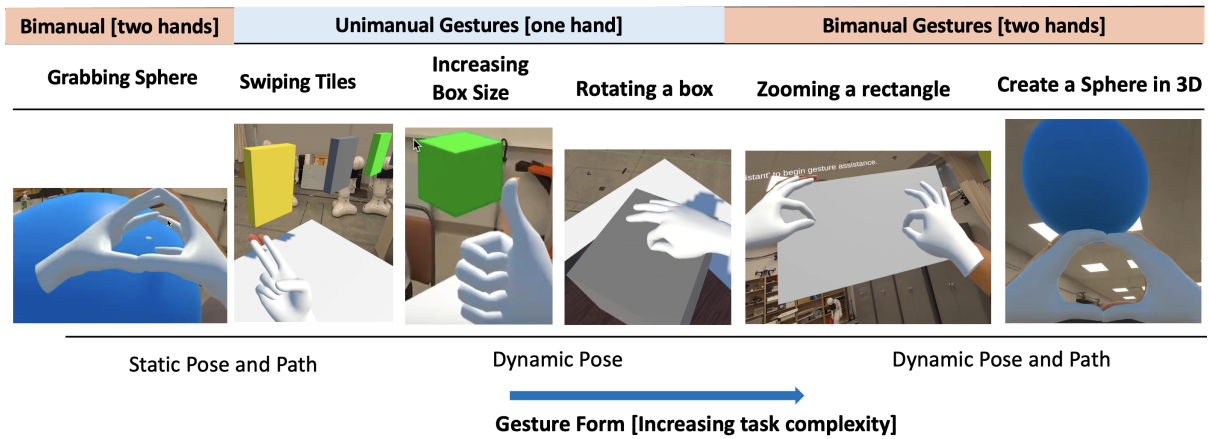


Fig. 4. Overview of Tasks and Gestures explored in Study 2 (User Evaluation): (a) Static Pose-and-Path (“Swiping tiles” as before) and an additional “Grabbing Sphere” task where users need to perform circle gesture with both hands and move it horizontally to manipulate the sphere. (b) Dynamic Pose (“Rotating a box” as before) and an additional task of “Increasing Box Size” where users need to perform thumbs up gesture with lateral thumb movements to scale the box (c) Bimanual Dynamic Pose-and-Path (“Zooming a rectangle” as before) and an additional task of “Create a Sphere in 3D” where users need to perform a circle gesture with both hands, fingertips touching and rotating this gesture clockwise to manipulate the sphere’s orientation.

Pose-and-Path, Dynamic Pose and Dynamic Pose-and-Path), starting first with the tasks involving unimanual and then bimanual gestures, with the order of the AR assistants randomized. Each task was presented one by one and allotted 2 minutes (6 minutes total per condition). To assess how AR assistants supported the accurate mental model formation, participants were asked to perform each task on their own before using the AR assistant. Subsequently, participants interacted with each AR assistant and were asked to think aloud how they would perform each task in AR after using each AR assistant. Participants were allowed to access help resources available on the Meta Quest 3 device (e.g., [49]) before or during the AR tasks. However, consistent with prior findings that users often skip tutorials [35], we did not force them to watch a specific tutorial, as we were interested in understanding the value of the ARGESTUREAID approach.

After completing all tasks for an assistant, participants were asked to complete a post-task questionnaire to evaluate their overall experience with AR assistant, focusing on discoverability, and usefulness when discovering and executing gestures. Lastly, follow-up interviews were conducted to probe into any difficulties that participants faced and their overall perceptions of AR assistants.

5.4 Data Collection and Analysis

We recorded each participant’s gestures and interactions during the study through multiple cameras, screen and audio recorded their interview responses. Our analysis focused on 1): how participants used AR assistance from gesture discovery to execution, and 2) how the assistants supported accurate mental model formation.

To assess differences between conditions, we conducted a Pearson’s Chi-square test of independence between AR Assistant Interventions (ARGESTUREAID vs. BASELINE AR ASSISTANT) and user responses (five-point Likert scale from Strongly Agree to Strongly Disagree). Task completion time (refers to total time participants took to complete the task) was divided into gesture discovery (from task prompt to gesture identification) and gesture execution (from gesture initiation to successful task completion). We manually annotated interaction logs to

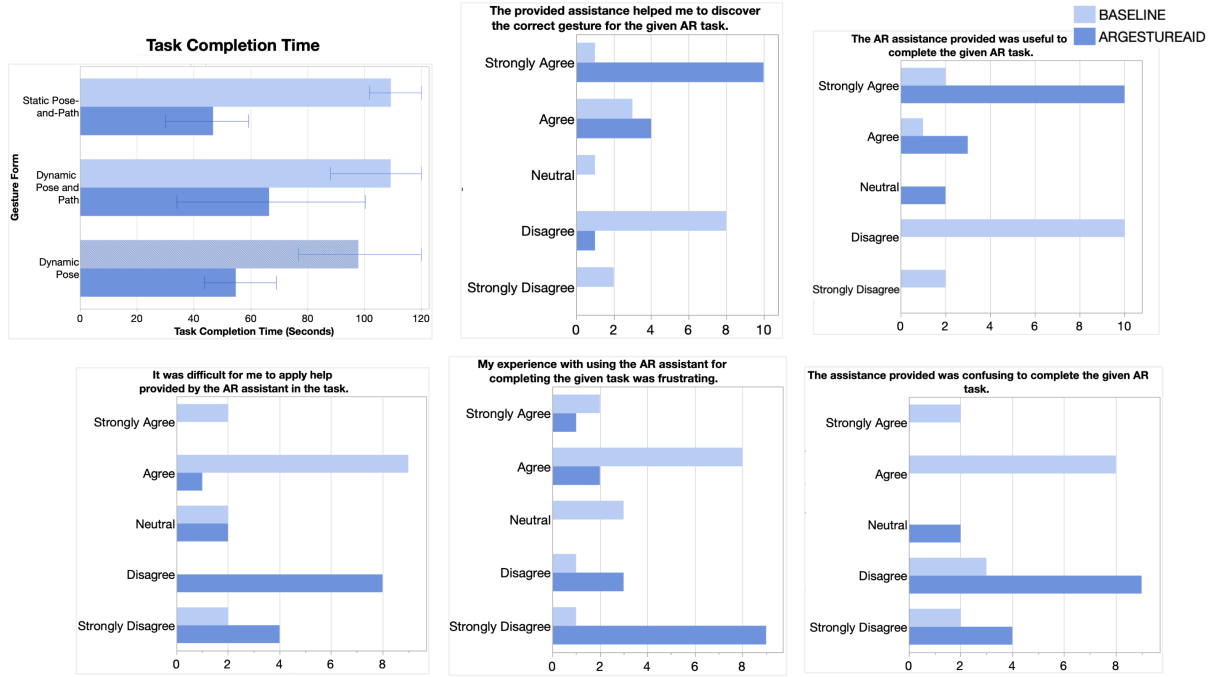


Fig. 5. Overview of task completion time and participants' responses to the post-task questionnaire. The Pearson Chi-Squared test showed a significant difference for each metric across both AR Assistant interventions for completing the tasks. With ARGESTUREAID, users demonstrated lower task completion time and indicated that ARGESTUREAID helped them discover and execute the correct gesture compared to BASELINE AR ASSISTANT.

examine trial-and-error strategies, identifying repeated or varied gesture attempts, non-linear task paths, and corrective sequences [18]. To complement quantitative findings, we analyzed think-aloud data and follow-up interviews using an inductive approach [22] and affinity diagramming to identify recurring themes related to discoverability, and usefulness.

5.5 Results

We have organized our results around key themes, discoverability and usefulness, to assess each AR assistant's performance. Overall, ARGESTUREAID supported novices in discovering and executing mid-air gestures more efficiently than the BASELINE AR ASSISTANT within the constraints of our controlled lab session. Participants completed tasks faster and rated ARGESTUREAID as more useful and discoverable, particularly when ongoing, gesture-aware feedback reduced trial-and-error attempts. However, precise execution of complex bimanual gestures remained challenging, highlighting opportunities to further improve adaptive guidance strategies.

5.5.1 Task Completion Time and Perceptions of AR Assistant Interventions.

Task Completion Time: On average, participants took significantly longer to complete tasks with BASELINE AR ASSISTANT (mean = 105s; Static Pose-and-Path = 89.4s, Dynamic Pose = 98s, Dynamic Pose-and-Path = 108s) than with ARGESTUREAID (mean = 56 s; Static = 46.8s, Dynamic = 55s, Dynamic Pose-and-Path = 66.5s) across all tasks and gestures. A paired-sample t-test revealed a significant difference ($t(82) = 9.32, p < 0.0001$, two-tailed). Most participants (12/15) could not complete all tasks with BASELINE AR ASSISTANT, especially bimanual dynamic

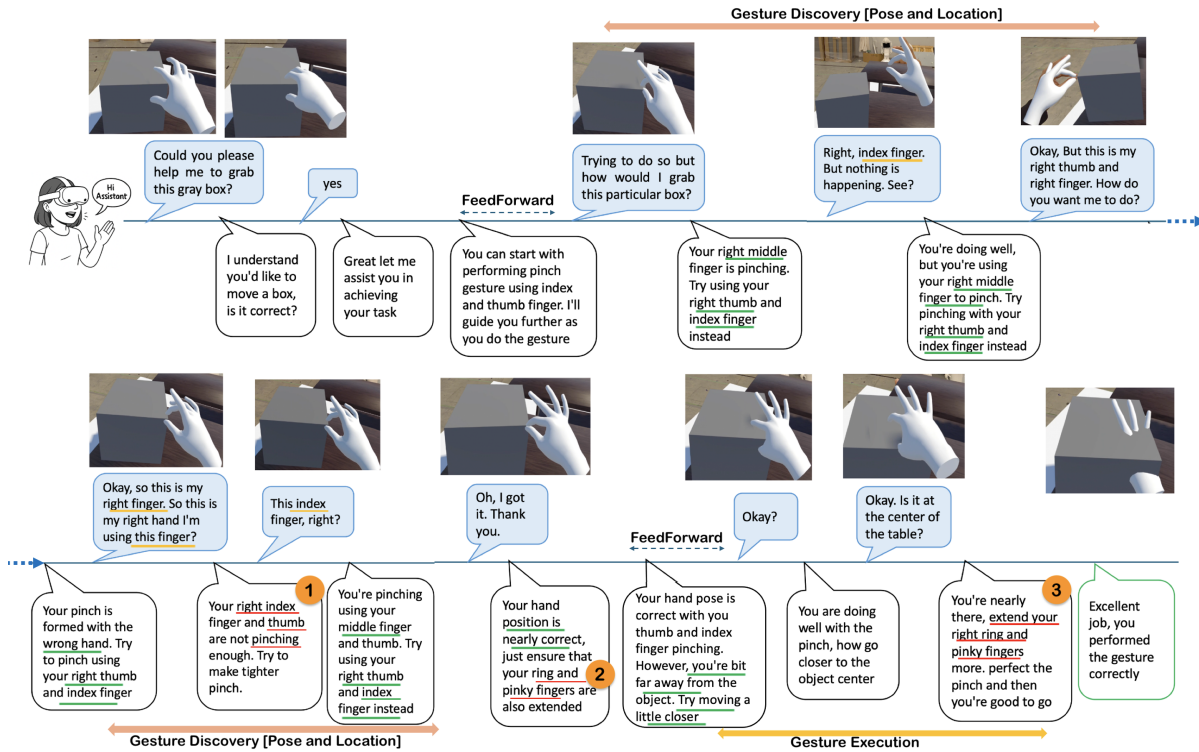


Fig. 6. Illustrative case of gesture discovery and execution for a unimanual dynamic pose-and-path gesture. The timeline shows user utterances and ARGESTUREAID's assistance across discovery and execution phases. Feedforward guidance appears during discovery, followed by corrective feedback during execution. Green highlights accurate assistance; red highlights incorrect or incomplete guidance. Yellow highlights trial-and-error attempts in user utterances. Numbered markers (1–3) indicate instances where assistance did not explicitly address the detected error.

pose-and-path gestures; of these, 7/12 discovered static gestures and 5/12 discovered dynamic pose gestures but failed in their execution. In contrast, nearly all participants (13/15) completed every task with ARGESTUREAID, except two who struggled with bimanual dynamic pose-and-path gestures.

Participants Valued Timely, Gesture-Aware Feedback: Most participants (13/15) perceived ARGESTUREAID's assistance as more useful than BASELINE AR ASSISTANT for completing AR tasks, and these results were significant ($\chi^2(4, N = 30) = 20.3, p = 0.0004$), and align with task completion scores. In particular, participants indicated that ARGESTUREAID's in-context, corrective assistance, which continuously monitored their gestures and dynamically adapted to their hand and finger positions to provide timely corrections when gestures were misaligned, was particularly useful compared to Baseline's generic LLM guidance that lacked such ongoing, gesture-aware feedback, as P12 commented: "As a first-time user, [ARGESTUREAID] did not overwhelm me with all the instructions...the actionable steps [that] tell me at the same moment what needs to be done...[the] live assistance was really helpful. I liked that [ARGESTUREAID] tracked how my hand and fingers are positioned and walked me through each step one at a time, okay, you do pinch first or move your hand to the box center, and then corrected me if my gesture was wrong. If it didn't tell me I wouldn't have known I need to move to the box center." (P12)

Generic, Tutorial-Like Guidance Was Hard to Follow: Most users (12/15) found BASELINE AR ASSISTANT's static, tutorial-like instructions to be redundant and not useful. P01 commented: "the assistant [BASELINE AR

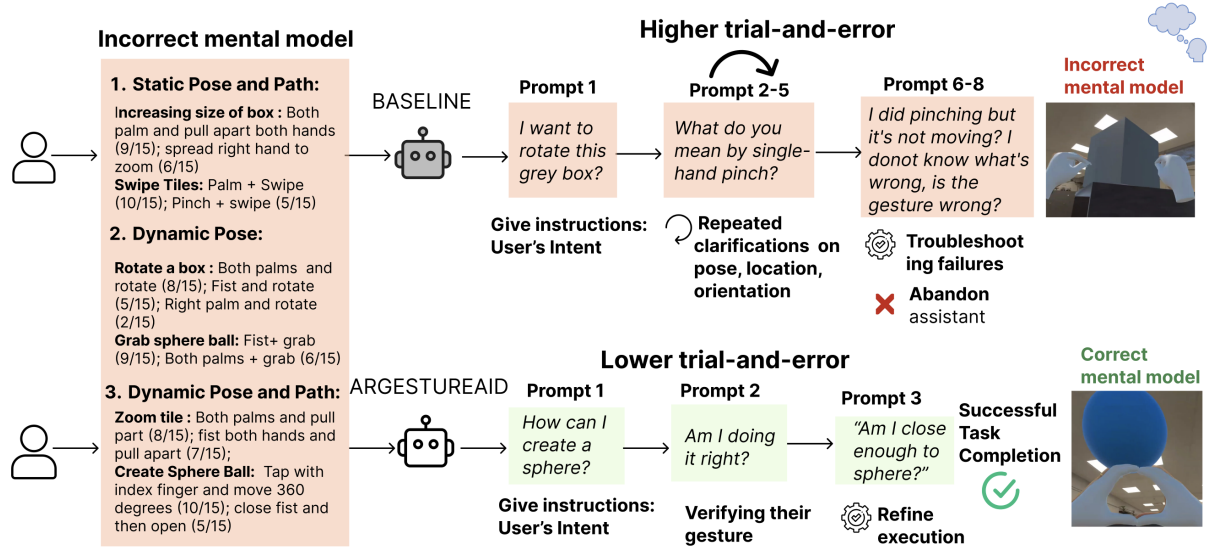


Fig. 7. **Trial-and-Error and Mental Model Differences in ARGESTUREAID vs BASELINE AR ASSISTANT:** This figure illustrates incorrect gestures attempted by all participants for each task before interacting with either assistant, reflecting incorrect mental models. We also illustrate the case study of Participant P13, who encountered higher trial-and-error with BASELINE AR ASSISTANT, primarily focused on repeated clarifications on gesture pose and debugging the BASELINE AR ASSISTANT's guidance, which ultimately led the user to abandon BASELINE AR ASSISTANT. In contrast, P13 experienced fewer trial-and-error instances with ARGESTUREAID, mostly related to verifying the correction of their gesture and refining execution. The user was able to complete the task successfully with ARGESTUREAID and formed correct mental model eventually.

ASSISTANT] was like reading a manual...there was no live feedback and it was just repeating a set of instructions that I actually had to know, but I was not able to implement them all at once. I needed a bit more help [from the assistant] when I was trying it [gesture] and what I was doing right or wrong.” (P01) Furthermore, participants expressed that BASELINE AR ASSISTANT's assistance was not relevant when they experienced gesture errors due to precise finger configuration, hand orientation, or movement timing or spatial requirements (e.g., where to pinch relative to objects). Users were frustrated with the repetitive, generic guidance that did not adapt across different tasks, as P15 commented: “[BASELINE AR ASSISTANT] was repeatedly giving me the same context for the first task, and even for the last. As a first-time user to AR headsets, I think giving more detailed instructions would be helpful, such as if there is a specific location...should I pinch inside or outside the box, or which fingers to use for pinching or how to rotate my wrist...” (P15)

Overall, most users preferred ARGESTUREAID over BASELINE AR ASSISTANT across all key measures, as summarized in Figure 5. Taken together, these findings indicate that providing feedback grounded in users' current gesture states helps limit repeated trial-and-error, as users receive specific guidance on how to adjust their actions rather than repeatedly reattempting the gesture based on static instructions.

5.5.2 Impact of AR Assistant Interventions on Users' Mental Models. We next evaluated how participants discovered the correct gesture using both of the AR assistants, focusing on the impact on users' mental models.

Gesture-Aware Assistance Supported Gesture Discovery Analysis of participants' gesture trials revealed that all participants initially performed incorrect gestures for each task before interacting with either assistant, reflecting incorrect mental models stemming from unfamiliarity with the correct gesture (see Figure 7). However,

after engaging with ARGESTUREAID, all participants successfully discovered the correct gestures across all tasks. In contrast, with BASELINE AR ASSISTANT, only 3/15 participants were able to discover the correct gestures. Most participants (14/15) perceived ARGESTUREAID as more helpful for discovering the correct gesture compared to BASELINE AR ASSISTANT and this difference was significant ($\chi^2(4, N = 30) = 15.9, p = 0.0031$).

Feedback Helped Users Move Beyond Incorrect Mental Models Before seeking help, users often tried incorrect gestures (e.g., making a fist; see Figures 7, 6). Many found mid-air gestures not transferable from real-world interactions, as P10 noted: *“It [mid-air gesture] isn’t natural, doing pinch or circle...I’ve never done in my day-to-day life.”* (P10) In these situations, ARGESTUREAID’s proactive feedforward redirected users from random trial-and-error toward the intended gesture during both discovery and execution. As P05 explained: *“I had thought it would be just picking the box [using both palms and rotate] as naturally done in day-to-day life, but it was not...So then, when it [ARGESTUREAID’s Feedforward] told me to pinch with index and middle finger, I realized that my gesture was wrong.”* (P05) This type of guidance prevented users from becoming stuck in incorrect strategies early on, an issue reported in prior work [34].

Furthermore, 11/15 participants expressed that ARGESTUREAID’s corrective feedback, provided at a granular level, grounded in the formal representation, and addressing specific aspects of their gestures, including their pose, orientation, location (e.g., wrong finger position, hand orientation, or movement timing) during discovery, as well as spatial distance relative to virtual environments (e.g., how far they needed to be from a virtual object) during execution, helped them to quickly diagnose and correct gesture errors: *“...precise assistance just like feedback from the second one [ARGESTUREAID] saying you are pinching all fingers [which fingers to use], fully extend middle, ring, and pinky [how much to extend them]. That clarified everything and helped me a lot to discover the correct gesture.”* (P08) Similarly, P05 commented: *“I liked the intermediate guidance that [ARGESTUREAID] gave me, which was really helpful when I was executing the gesture...it was confirming whether I was going to the right place on the tile, where I was supposed to be in the 3D space to execute the function [gesture] and perform the movement properly.”* (P05)

Furthermore, some participants (6/15) appreciated feedback on both the correct and incorrect aspects of the gesture, as it helped them understand which steps they had already performed successfully. As P10 commented: *“...both aspects of [ARGESTUREAID’s assistance], whether I was doing it right or not, such as being told that my pinch gesture was correct, were extremely helpful in confirming that I was on the right track and in helping me discover the gesture.”* (P10)

In contrast, participants reported that BASELINE AR ASSISTANT provided only surface-level instructions which did not help in discovery, as users perceived multiple possible interpretations of gestures but received no clarification about which specific variation was required, leaving them confused, as P07 commented: *“Because there are many interpretations [of a gesture], like a pinch, I was very confused. It [BASELINE AR ASSISTANT] was not clear...you can pinch with all fingers or index or middle finger and that’s how it’s not simple for you. I didn’t understand where my fingers were supposed to be in 3D space...Should I mirror the gesture, or do the same thing with both hands?”* (P07)

Gesture-Aware Feedback Reshaped Trial-and-Error Participants made nearly twice as many trial-and-error queries with BASELINE AR ASSISTANT (298) as with ARGESTUREAID (147) across discovery and execution phases (Table 3, Figure 7). Over half of BASELINE AR ASSISTANT queries (165/298) were repeated clarifications about gesture pose, location, orientation (e.g., *“What do you mean by single-hand pinch?”* (P04), *“Do I pinch the box and then move it, or pinch and then move?”* (P02)). Many participants expressed frustration that the assistant seemed unaware of their live hand movements. The rest involved troubleshooting failures (86/298) or reflected incorrect mental models (47/298), such as assuming a pinch would trigger a swipe gesture (e.g., *“How can I do pinch gesture correctly to move these tiles?”*).

In contrast, over half of the queries with ARGESTUREAID (89/147) focused on verifying whether gestures were performed correctly (See Figure 7, Table 3). Of these, 66 concerned verifying pose and location during gesture discovery (e.g., *“Is this index pinch correct?”*), while 33 focused on gesture placement during execution (e.g., *“Am*

Table 3. Comparison of user utterance types during trial-and-error interactions: ARGESTUREAID vs. BASELINE AR ASSISTANT.

ARGESTUREAID (N=147)		BASELINE AR ASSISTANT (N=298)	
Utterance Type	Examples	Utterance Type	Examples
Verifying the correct pose and location (89/147)	<i>"Is it correct?", "Am I close enough to the box?"</i>	Repeated clarifications on gesture pose, location and orientation (165/298)	<i>"What do you mean by single-hand pinch", "That's not working, do I pinch the box and then move it, or pinch and then move?"</i>
Follow-ups regarding prior assistance, such as finger use or gesture execution (31/147)	<i>"Is the index finger the second one?", "Which way do I pinch?"</i>	Troubleshooting failures (86/298)	<i>"I did pinching but it's not moving? I donot know what's wrong, is the gesture wrong?", "I donot know how to use both my hands, is it mirrored or is it suppose to be same thing?"</i>
Queries for correct execution (27/147)	<i>"How close should I come to the object?", "How do I increase the speed of the zoom?"</i>	Queries stemmed from incorrect mental models (47/298)	<i>"How can I do pinch gesture correctly to move these tiles?"</i> (assuming a pinch would work for a swipe)

I close enough to the box?"). A few non-native English speakers (4/15) also struggled with finger terminology, particularly during complex dynamic or bimanual tasks. During discovery, 31/159 trials involved clarification questions (e.g., *"I'm not an english speaker, what is [the] index finger?"*), and during execution, 27/147 trials questions about proximity (*"How close should I come to the object?"*) or adjusting gesture speed (*"How do I increase the speed of the zoom?"*)

Overall, most participants (9/15) were enthusiastic about using ARGESTUREAID for future AR tasks, as P07 commented: *"I really appreciate [ARGESTUREAID], especially for a new user and non-English native speaker, [ARGESTUREAID's] ability to see my finger and to find what's wrong with the real gestures, providing concrete guidance, paying attention to what I did...and helping me with terms (e.g., index finger) which I'm not familiar, doing something a teacher usually do"* (P07). However, few users expressed when assistance was unclear (e.g., understanding finger terminologies, how much to rotate or how to orient the hands front or back), especially for gestures beyond simple pinch actions (e.g., bimanual or static pose-and-path gestures), more explicit visual aids in the form of arrows showing hand movements could be beneficial. As P05 commented: *"In the [swiping] task... it was difficult to me to understand that my hands should be palm facing the floor and the object. For someone not used to perform the gestures visual clue or arrows along with audio indicating the [hand] movement that I should follow would have been very helpful."* (P05)

6 DISCUSSION

Our design and initial findings with ARGESTUREAID highlight the promise of in-context conversational assistance for mid-air gestures, where guidance is grounded in a structured, interpretable representation of users' task intent and current gesture state. ARGESTUREAID uses this representation to compare what users intend to do with how their gestures are currently executed, enabling targeted, error-aware feedforward and corrective feedback during gesture discovery and execution. In our studies, participants described the value of receiving timely, specific prompts when their pose or execution diverged from what the task required, particularly in

Table 4. Design considerations and implications synthesized from our observations for in-context conversational assistance during mid-air gesture discovery and execution.

Design Consideration	Key Implications
Structured gesture representations enable semantic reasoning	Interpretable gesture features (e.g., pose, orientation, hand–object relationships) allow conversational AR assistants to reason about user intent, partial success, and errors, and support reuse across tasks and gestures.
LLMs are most effective when used as semantic mediators	Interpreting structured gesture state relative to task goals (rather than generating generic instructions) enables flexible, context-aware mappings from low-level kinematic input to meaningful, situation-specific guidance
Feedforward is especially valuable early in interaction	Early feedforward supports the formation of initial mental models of gesture form and spatial requirements during gesture discovery.
Combining feedforward and corrective feedback shapes exploration	Grounding both feedforward and corrective feedback in error-prone gesture features (e.g., pose, orientation, position) helps reduce unstructured trial-and-error and supports recovery during execution.
Error-aware feedback should balance correction and reassurance	Feedback that highlights both incorrect and correct aspects of a gesture helps users adjust their actions while maintaining confidence.
Conversational feedback benefits from spatial grounding	While audio guidance supports hands-free interaction, gesture placement often involves spatial ambiguities that are difficult to convey verbally and may benefit from lightweight visual cues.
Formal gesture representations enable selective automation	Explicit gesture representations can support selective automation during discovery while preserving user control during execution.

moments of uncertainty. Building on these observations, we distill a set of design considerations for gesture-aware conversational assistants in immersive AR (Table 4).

6.1 Leveraging Formal Representations: Bridging Gesture Recognition and LLM Conversational Intelligence

Traditional rule-based systems [20, 67, 75] rely on fixed mappings from kinematic features to feedback, requiring manually authored rules for each gesture and task, and they struggle with ambiguity, partial success, and task-dependent interpretations of errors (e.g., “you’re almost there, rotate your wrist slightly”). In contrast, our representation-centric approach uses LLMs to reason over structured gesture states and task goals, enabling flexible mappings from continuous kinematic input to meaningful, situation-specific guidance. Rather than treating the LLM as a surface-level text generator, ARGESTUREAID transforms interpretable gesture descriptors, such as pose, orientation, and object-relative spatial relationships—into a form suitable for semantic reasoning. This design supports more reliable identification and correction of gesture errors and aligns with broader trends in HCI and ML showing that structured representations can improve the robustness and interpretability of generative model outputs [7].

Building on this representation-centric approach, the LLM can adapt feedback to partial success, near misses, or ambiguous input—capabilities that are difficult to achieve with fixed, rule-based systems. This approach allows ARGESTUREAID to interpret gestures relative to task goals, recognizing when the same hand pose may be appropriate in one context but erroneous in another. By focusing on error-prone gesture features such as pose, orientation, and object-relative spatial relationships, the system can provide more precise, situation-specific feedback, including for spatially complex and bimanual gestures. The same design also supports extensibility, as

new gestures or tasks can be introduced by updating the structured representation and prompt examples rather than rewriting feedback logic.

Advances in GenAI copilots for complex software tasks [2, 9, 61] suggest opportunities to explore varying levels of automation in mid-air gesture interaction. Formal gesture representations could help determine which aspects of gesture learning benefit from full automation versus step-by-step guidance. For example, gesture discovery could be automated, so users no longer need to explore repetitive poses or orientations. Once the system identifies a user's intent, a 3D visual avatar of the correct gesture can appear, allowing users to observe and interact with it, as P07 noted, “...a visual avatar that can do the gesture for me and show me the process I'm following.” (P07) For execution, the copilot can provide context-aware guidance, letting users perform gestures relative to virtual objects using simple inputs like taps or controllers, maintaining control while reducing complexity. Future research could explore such semi-automated copilots [36] within AR environments that balance automation and guidance to demonstrate, scaffold, and gradually transfer control to users during gesture learning.

6.2 Feedforward vs Feedback Grounded in Formal Representation

Our findings suggest that ongoing corrective guidance combined with iterative feedforward–feedback loops, structured according to a formal representation of error-prone features (e.g., pose, orientation, and relative positioning of gestures to virtual objects) reduces random gesture trial-and-error before interaction begins and assists users in discovering and executing the correct gestures when they encounter errors. By grounding both feedforward and feedback in the same formal gesture representation, ARGESTUREAID maintains consistency between what the system anticipates, detects, and explains to the user. Notably, participants could even discover, execute bimanual gestures and complete complex tasks within lower task completion times using ARGESTUREAID compared to BASELINE AR ASSISTANT. Consistent with prior research on mental models (e.g., [35, 70]) in emerging interaction paradigms, our study highlights the importance of providing support for precise hand configurations and positioning relative to virtual objects through ongoing assistance. In line with earlier findings [50], feedforward cues in ARGESTUREAID oriented users to the correct gesture discovery (e.g., pose, position, orientation, and location) for their intended tasks. By providing explicit, on-the-fly guidance, feedforward reduced “random” gestures or human errors (as observed in earlier studies [35]) stemming from experimentation and incorrect mental models. Consequently, participants could devote more of their help-seeking efforts to achieving the correct pose for their task rather than struggling with trial-and-error.

Corrective feedback, on the other hand, offered context-dependent error explanations. Even novice participants who initially struggled to determine hand positions or gestures completed challenging AR tasks when they received real-time hints about “where” and “why” they were making mistakes. Consistent with prior work on explainable chatbot interfaces [34], participants valued feedback that not only pinpointed *what went wrong* but also *what was being done correctly*, providing both guidance and reassurance. Although ARGESTUREAID accurately detected gesture-formation errors, participants sometimes struggled to interpret its feedback. Ambiguous instructions, such as “move closer” without explicit reference boundaries of the virtual object, illustrate the challenge of providing guidance that is both accurate and interpretable. Drawing inspiration from Explainable AI [28, 44, 54], future research can focus on phrasing, specificity, and contextual cues of feedback, for example by referencing object edges or centers, to help users understand the error and how to correct it while maintaining confidence and agency in gesture execution.

6.3 Multimodal Feedback

While audio guidance proved beneficial overall, users highlighted the need of subtle visual feedback along with audio, especially at instances when figuring out the location of the gesture. Specifically, providing arrow-based overlays that indicate the exact hand movement for a pinch or grasp could be beneficial, as expressed by P10:

“If arrows appeared in my hands showing the hand movement that I should follow, that visual clue would be very helpful along with audio.” Future work could explore when vocal corrective guidance is more effective than static visual cues. As devices move toward screenless interactions, consistent with the literature on gesture discovery and corrective assistance approaches [20], unobtrusive yet informative multimodal cues, combining visual and audio-based assistance, can reduce the guesswork associated with positioning and orientation. However, our current prototype exhibited occasional latency, which undermined the immediacy of these cues and, in turn, affected users’ interaction with the assistant. To address this challenge, future AR gesture-assistance systems could integrate low-latency edge computing or more localized LLM processing, thereby improving real-time responsiveness and ensuring that subtle, context-sensitive guidance remains consistently available and effective.

7 LIMITATIONS

While we expanded our implementation to include multiple mutually exclusive gesture types supported by commercial AR/VR systems, our evaluation was limited to a finite set of tasks and gestures on a single commercial AR headset with integrated hand tracking and microphones. Although our approach can, in principle, be applied to other AR platforms capable of capturing hand-sensing data, further studies are needed to assess its performance across diverse devices and sensing modalities (e.g., smart glasses or surface electromyography [38]), where differences in tracking fidelity may affect the quality of in-context assistance. In addition, our short-term lab study focused on examining feasibility and comparative strengths and weaknesses of ARGESTUREAID relative to BASELINE AR ASSISTANT across a range of gesture types and interaction scenarios. Longitudinal and in-the-wild studies are needed to surface practical challenges that may emerge over extended use, such as user fatigue, environmental noise, or dynamic obstacles that influence hand positioning. Larger-scale field studies could also help clarify how user preferences—for example, the desired level of detail in audio-based guidance—evolve over time.

8 CONCLUSIONS

We presented ARGESTUREAID, a proof-of-concept conversational assistant that combines users’ voice-based task descriptions with structured representations of hand gesture data and LLM-based reasoning to provide in-context feedforward and corrective feedback. By grounding assistance in users’ task intent and current gesture state, ARGESTUREAID supports both gesture discovery and execution through guidance that adapts to observed errors and interaction context. Across two studies, our findings indicate that participants found ARGESTUREAID more supportive for gesture discoverability, execution, and perceived usefulness than a general-purpose conversational assistant without access to gesture state. Together, these results suggest that adaptive, in-context feedforward and corrective feedback grounded in formal gesture representations and observed error patterns may provide meaningful support during mid-air gesture interaction. Beyond the specific system implementation, this work highlights the value of grounding conversational AR assistance in structured, interpretable gesture representations. We synthesize a set of design considerations based on our observations that can inform future work on gesture-aware, context-sensitive assistance for immersive AR systems.

Acknowledgments

We thank Meta Inc. for funding this research. We also thank Michael Glueck and Ben Lafreniere for their valuable feedback in the early stages of this research.

References

- [1] 2020. Get holomeeting 2. <https://www.microsoft.com/en-us/p/holomeeting-2/9nh5zdb4ggbq?activetab=pivot%3Aoverviewtab> [Accessed 21-04-2026].
- [2] 2024. Figma AI: Your Creativity, unblocked with Figma AI — figma.com. <https://www.figma.com/ai/>. [Accessed 11-09-2024].

- [3] 2025. ChatGPT Voice mode — chatgpt.com. <https://chatgpt.com/features/voice/>. [Accessed 30-10-2025].
- [4] 2025. Learn more about Meta AI on Meta Quest | Quest Help | Meta Store — meta.com. https://www.meta.com/help/quest/1008196844074415/?srsltid=AfmBOorUaGyNaf_fDGJKAzsM297_c3XjrTp1KsvfW2v-Vxuc45Af3ILC. [Accessed 30-10-2025].
- [5] 2025. OpenAI Platform — platform.openai.com. <https://platform.openai.com/docs/guides/realtime-vad>. [Accessed 30-10-2025].
- [6] 2025. OVRSkeleton Class | Unity: Core SDK v69 — developers.meta.com. https://developers.meta.com/horizon/reference/unity/v69/class_o_v_r_skeleton/. [Accessed 23-04-2025].
- [7] Sabri My Abdelouahed, Rahmouni Abba, Ennaji Asmae, and Aarab Abdellah. 2024. Harnessing feature engineering to improve machine learning: A review of different data processing techniques. In *2024 International Conference on Intelligent Systems and Computer Vision (ISCV)*. 1–6. doi:10.1109/ISCV60512.2024.10620105
- [8] Gregory D. Abowd and Alan J. Dix. 1992. Giving undo attention. *Interacting with Computers* 4, 3 (12 1992), 317–342. arXiv:<https://academic.oup.com/iwc/article-pdf/4/3/317/2175174/iwc4-0317.pdf> doi:10.1016/0953-5438(92)90021-7
- [9] Adobe. 2023. Adobe unveils Firefly, a family of new creative generative ai. <https://news.adobe.com/news/news-details/2023/Adobe-Unveils-Firefly-a-Family-of-new-Creative-Generative-AI/default.aspx> [Accessed 21-09-2024].
- [10] Florian Alt, Sabrina Geiger, and Wolfgang Höhl. 2018. ShapelineGuide: Teaching Mid-Air Gestures for Large Interactive Displays. In *Proceedings of the 7th ACM International Symposium on Pervasive Displays (Munich, Germany) (PerDis '18)*. Association for Computing Machinery, New York, NY, USA, Article 3, 8 pages. doi:10.1145/3205873.3205887
- [11] Anthropic. 2024. Model Context Protocol (MCP) - Claude Docs — docs.claude.com. <https://docs.claude.com/en/docs/mcp>. [Accessed 30-10-2025].
- [12] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2012. Dwell-and-spring: undo for direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (Austin, Texas, USA) (CHI '12)*. Association for Computing Machinery, New York, NY, USA, 1957–1966. doi:10.1145/2207676.2208339
- [13] Olivier Bau and Wendy E. Mackay. 2008. OctoPocus: a dynamic guide for learning gesture-based command sets. In *Proceedings of the 21st Annual ACM Symposium on User Interface Software and Technology (Monterey, CA, USA) (UIST '08)*. Association for Computing Machinery, New York, NY, USA, 37–46. doi:10.1145/1449715.1449724
- [14] Mark Billinghurst. 2021. Grand Challenges for Augmented Reality. *Frontiers in Virtual Reality* 2 (2021). doi:10.3389/frvir.2021.578080
- [15] Mark Billinghurst, Adrian Clark, Gun Lee, et al. 2015. A survey of augmented reality. *Foundations and Trends® in Human-Computer Interaction* 8, 2-3 (2015), 73–272.
- [16] Lal Bozgeyikli, Andrew Rajj, Srinivas Katkoori, and Redwan Alqasemi. 2017. Effects of instruction methods on user experience in virtual reality serious games. In *Virtual, Augmented and Mixed Reality: 9th International Conference, VAMR 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings 9*. Springer, 215–226.
- [17] Gavin Buckingham. 2021. Hand Tracking for Immersive Virtual Reality: Opportunities and Challenges. *Frontiers in Virtual Reality Volume 2 - 2021* (2021). doi:10.3389/frvir.2021.728461
- [18] Donald T Campbell. 1960. Blind variation and selective retentions in creative thought as in other knowledge processes. *Psychological review* 67, 6 (1960), 380.
- [19] John M. Carroll and Mary Beth Rosson. 1987. *Paradox of the Active User*. MIT Press, Cambridge, MA, USA, 80–111.
- [20] Edwige Chauvergne, Martin Hachet, and Arnaud Prouzeau. 2023. User Onboarding in Virtual Reality: An Investigation of Current Practices. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (Hamburg, Germany) (CHI '23)*. Association for Computing Machinery, New York, NY, USA, Article 711, 15 pages. doi:10.1145/3544548.3581211
- [21] Philip R. Cohen, Michael Johnston, David McGee, Sharon Oviatt, Jay Pittman, Ira Smith, Liang Chen, and Josh Clow. 1997. QuickSet: multimodal interaction for distributed applications. In *Proceedings of the Fifth ACM International Conference on Multimedia (Seattle, Washington, USA) (MULTIMEDIA '97)*. Association for Computing Machinery, New York, NY, USA, 31–40. doi:10.1145/266180.266328
- [22] Juliet M Corbin and Anselm Strauss. 1990. Grounded theory research: Procedures, canons, and evaluative criteria. *Qualitative sociology* 13, 1 (1990), 3–21.
- [23] Justin Cranshaw, Emad Elwany, Todd Newman, Rafal Kocielnik, Bowen Yu, Sandeep Soni, Jaime Teevan, and Andrés Monroy-Hernández. 2017. Calendar.help: Designing a Workflow-Based Scheduling Agent with Humans in the Loop. *ACM CHI Conference on Human Factors in Computing Systems* (January 2017). <https://www.microsoft.com/en-us/research/publication/calendar-help-designing-workflow-based-scheduling-agent-humans-loop/>
- [24] Nem Khan Dim, Chaklam Silpasuwanchai, Sayan Sarcar, and Xiangshi Ren. 2016. Designing Mid-Air TV Gestures for Blind People Using User- and Choice-Based Elicitation Approaches. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems (Brisbane, QLD, Australia) (DIS '16)*. Association for Computing Machinery, New York, NY, USA, 204–214. doi:10.1145/2901790.2901834
- [25] Daniel Eckhoff, Jan Schnupp, and Alvaro Cassinelli. 2024. Temporal precision and accuracy of audio-visual stimuli in mixed reality systems. *PLOS ONE* 19, 1 (01 2024), 1–15. doi:10.1371/journal.pone.0295817
- [26] Margarita Esau, Veronika Krauß, Dennis Lawo, and Gunnar Stevens. 2022. Losing Its Touch: Understanding User Perception of Multimodal Interaction and Smart Assistance. In *Designing Interactive Systems Conference (Virtual Event, Australia) (DIS '22)*. Association for Computing Machinery, New York, NY, USA, 1288–1299. doi:10.1145/3532106.3533455

- [27] Yahya Ghazwani and Shamus Smith. 2020. Interaction in Augmented Reality: Challenges to Enhance User Experience. In *Proceedings of the 2020 4th International Conference on Virtual and Augmented Reality Simulations* (Sydney, NSW, Australia) (ICVARs 2020). Association for Computing Machinery, New York, NY, USA, 39–44. doi:10.1145/3385378.3385384
- [28] Alyssa Glass, Deborah L. McGuinness, and Michael Wolverton. 2008. Toward Establishing Trust in Adaptive Agents. In *Proceedings of the 13th International Conference on Intelligent User Interfaces* (Gran Canaria, Spain) (IUI '08). Association for Computing Machinery, New York, NY, USA, 227–236. doi:10.1145/1378773.1378804
- [29] Ari Grobman. 2018. AR's success depends on perfecting input methods. <https://thenextweb.com/news/ars-success-depends-on-perfecting-input-methods>
- [30] Alexander Hoffmann, Daniel Spelmezan, and Jan Borchers. 2009. TypeRight: a keyboard with tactile error prevention. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA) (CHI '09). Association for Computing Machinery, New York, NY, USA, 2265–2268. doi:10.1145/1518701.1519048
- [31] John B. Horrigan and Sydney Jones. 2020. When technology fails. <https://www.pewresearch.org/internet/2008/11/16/when-technology-fails/>
- [32] Masoumehsadat Hosseini, Tjado Ihmels, Ziqian Chen, Marion Koelle, Heiko Müller, and Susanne Boll. 2023. Towards a Consensus Gesture Set: A Survey of Mid-Air Gestures in HCI for Maximized Agreement Across Domains. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems* (Hamburg, Germany) (CHI '23). Association for Computing Machinery, New York, NY, USA, Article 311, 24 pages. doi:10.1145/3544548.3581420
- [33] Anjali Khurana, Parsa Alamzadeh, and Parmit K. Chilana. 2021. ChatrEx: Designing Explainable Chatbot Interfaces for Enhancing Usefulness, Transparency, and Trust. In *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 1–11. doi:10.1109/VL/HCC51201.2021.9576440
- [34] Anjali Khurana, Parsa Alamzadeh, and Parmit K. Chilana. 2021. ChatrEx: Designing Explainable Chatbot Interfaces for Enhancing Usefulness, Transparency, and Trust. In *2021 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 1–11. doi:10.1109/VL/HCC51201.2021.9576440
- [35] Anjali Khurana, Michael Glueck, and Parmit K. Chilana. 2024. Do I Just Tap My Headset? How Novice Users Discover Gestural Interactions with Consumer Augmented Reality Applications. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 4, Article 165 (Jan. 2024), 28 pages. doi:10.1145/3631451
- [36] Anjali Khurana, Xiaotian Su, April Yi Wang, and Parmit K Chilana. 2025. Do It For Me vs. Do It With Me: Investigating User Perceptions of Different Paradigms of Automation in Copilots for Feature-Rich Software. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems* (CHI '25). Association for Computing Machinery, New York, NY, USA, Article 880, 18 pages. doi:10.1145/3706598.3713431
- [37] Anjali Khurana, Hariharan Subramonyam, and Parmit K Chilana. 2024. Why and When LLM-Based Assistants Can Go Wrong: Investigating the Effectiveness of Prompt-Based Interactions for Software Help-Seeking. In *Proceedings of the 29th International Conference on Intelligent User Interfaces* (Greenville, SC, USA) (IUI '24). Association for Computing Machinery, New York, NY, USA, 288–303. doi:10.1145/3640543.3645200
- [38] Ctrl labs at Reality Labs, David Sussillo, Patrick Kaifosh, and Thomas Reardon. 2024. A generic noninvasive neuromotor interface for human-computer interaction. *Biorxiv* (2024), 2024–02.
- [39] Morgan Le Chénéchal, Thierry Duval, Valérie Gouranton, Jérôme Royan, and Bruno Arnaldi. 2016. Vishnu: virtual immersive support for helping users an interaction paradigm for collaborative remote guiding in mixed reality. In *2016 IEEE Third VR International Workshop on Collaborative Virtual Environments (3DCVE)*. IEEE, 9–12.
- [40] JangHyeon Lee and Lawrence H Kim. 2024. Augmenting reality to diminish distractions for cognitive enhancement. *arXiv preprint arXiv:2403.03875* (2024).
- [41] Jaewook Lee, Sebastian S. Rodriguez, Raahul Natarajan, Jacqueline Chen, Harsh Deep, and Alex Kirlik. 2021. What's This? A Voice and Touch Multimodal Approach for Ambiguity Resolution in Voice Assistants. In *Proceedings of the 2021 International Conference on Multimodal Interaction* (Montréal, QC, Canada) (ICMI '21). Association for Computing Machinery, New York, NY, USA, 512–520. doi:10.1145/3462244.3479902
- [42] Jaewook Lee, Jun Wang, Elizabeth Brown, Liam Chu, Sebastian S. Rodriguez, and Jon E. Froehlich. 2024. GazePointAR: A Context-Aware Multimodal Voice Assistant for Pronoun Disambiguation in Wearable Augmented Reality. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) (CHI '24). Association for Computing Machinery, New York, NY, USA, Article 408, 20 pages. doi:10.1145/3613904.3642230
- [43] Kevin Li, Sachin Goyal, João D Semedo, and J Zico Kolter. 2024. Inference Optimal VLMs Need Fewer Visual Tokens and More Parameters. In *The Thirteenth International Conference on Learning Representations*.
- [44] Q Vera Liao and Jennifer Wortman Vaughan. 2023. AI Transparency in the Age of LLMs: A Human-Centered Research Roadmap. *arXiv preprint arXiv:2306.01941* (2023).
- [45] Tica Lin, Ben Lafreniere, Yan Xu, Tovi Grossman, Daniel Wigdor, and Michael Glueck. 2023. XR Input Error Mediation for Hand-Based Input: Task and Context Influences a User's Preference. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*.

- 1006–1015. doi:10.1109/ISMAR59233.2023.00117
- [46] Marco Lorigian. 2008. Undo for mobile phones: does your mobile phone need an undo key? do you?. In *Proceedings of the 5th Nordic Conference on Human-Computer Interaction: Building Bridges* (Lund, Sweden) (*NordiCHI '08*). Association for Computing Machinery, New York, NY, USA, 274–282. doi:10.1145/1463160.1463190
- [47] Ewa Luger and Abigail Sellen. 2016. "Like Having a Really Bad PA": The Gulf between User Expectation and Experience of Conversational Agents. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) (*CHI '16*). Association for Computing Machinery, New York, NY, USA, 5286–5297. doi:10.1145/2858036.2858288
- [48] Mathias N. Lystbæk, Peter Rosenberg, Ken Pfeuffer, Jens Emil Grønbaek, and Hans Gellersen. 2022. Gaze-Hand Alignment: Combining Eye Gaze and Mid-Air Pointing for Interacting with Menus in Augmented Reality. *Proc. ACM Hum.-Comput. Interact.* 6, ETRA, Article 145 (May 2022), 18 pages. doi:10.1145/3530886
- [49] Meta. 2022. Introducing "first hand," our official hand tracking demo built with presence platform's interaction SDK. <https://developer.oculus.com/blog/introducing-first-hand/>
- [50] Andreea Muresan, Jess McIntosh, and Kasper Hornbæk. 2023. Using Feedforward to Reveal Interaction Possibilities in Virtual Reality. *ACM Trans. Comput.-Hum. Interact.* 30, 6, Article 82 (Sept. 2023), 47 pages. doi:10.1145/3603623
- [51] Brad A Myers, David A Weitzman, Amy J Ko, and Duen H Chau. 2006. Answering why and why not questions in user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*. 397–406.
- [52] Donald Norman. 1990. A.(1990) The Design of Everyday Things.
- [53] Uran Oh, Shaun K. Kane, and Leah Findlater. 2013. Follow that sound: using sonification and corrective verbal feedback to teach touchscreen gestures. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility* (Bellevue, Washington) (*ASSETS '13*). Association for Computing Machinery, New York, NY, USA, Article 13, 8 pages. doi:10.1145/2513383.2513455
- [54] Martin Porcheron, Joel E. Fischer, Stuart Reeves, and Sarah Sharples. 2018. Voice Interfaces in Everyday Life. 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, 1–12. doi:10.1145/3173574.3174214
- [55] Shwetha Rajaram, Hemant Bhaskar Surale, Codie McConkey, Carine Rognon, Hrim Mehta, Michael Glueck, and Christopher Collins. 2025. Gesture and Audio-Haptic Guidance Techniques to Direct Conversations with Intelligent Voice Interfaces. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*. 1–20.
- [56] Ashwin Ramachandran and R Michael Young. 2005. Providing intelligent help across applications in dynamic user and environment contexts. In *Proceedings of the 10th international conference on Intelligent user interfaces*. 269–271.
- [57] Michael L Raulin and Anthony M Graziano. 2019. Quasi-experiments and correlational studies. In *Companion Encyclopedia of Psychology*. Routledge, 1122–1141.
- [58] Naveen Sendhilnathan, Ting Zhang, Ben Lafreniere, Tovi Grossman, and Tanya R. Jonker. 2022. Detecting Input Recognition Errors and User Errors using Gaze Dynamics in Virtual Reality. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (Bend, OR, USA) (*UIST '22*). Association for Computing Machinery, New York, NY, USA, Article 38, 19 pages. doi:10.1145/3526113.3545628
- [59] Soroush Shahi, Vimal Mollyn, Cori Tymoszek Park, Runchang Kang, Asaf Liberman, Oron Levy, Jun Gong, Abdelkareem Bedri, and Gierad Laput. 2024. Vision-Based Hand Gesture Customization from a Single Demonstration. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (Pittsburgh, PA, USA) (*UIST '24*). Association for Computing Machinery, New York, NY, USA, Article 54, 14 pages. doi:10.1145/3654777.3676378
- [60] Jungpil Shin, Akitaka Matsuoka, Md. Al Mehedi Hasan, and Azmain Yakin Srizon. 2021. American Sign Language Alphabet Recognition by Extracting Feature from Hand Pose Estimation. *Sensors* 21, 17 (2021). doi:10.3390/s21175856
- [61] Jared Spataro. 2023. Introducing Microsoft 365 copilot – your copilot for work. <https://blogs.microsoft.com/blog/2023/03/16/introducing-microsoft-365-copilot-your-copilot-for-work/>
- [62] 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*. 161–174.
- [63] Jo Vermeulen, Kris Luyten, Elise van den Hoven, and Karin Coninx. 2013. Crossing the Bridge over Norman's Gulf of Execution: Revealing Feedforward's True Identity. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) (*CHI '13*). Association for Computing Machinery, New York, NY, USA, 1931–1940. doi:10.1145/2470654.2466255
- [64] VRChat. 2025. VRChat — hello.vrchat.com. <https://hello.vrchat.com/>. [Accessed 22-04-2025].
- [65] Alexandra Vtyurina, Adam Fourney, Meredith Ringel Morris, Leah Findlater, and Ryen W. White. 2019. VERSE: Bridging Screen Readers and Voice Assistants for Enhanced Eyes-Free Web Search. In *Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility* (Pittsburgh, PA, USA) (*ASSETS '19*). Association for Computing Machinery, New York, NY, USA, 414–426. doi:10.1145/3308561.3353773
- [66] Gang Wang, Gang Ren, Xinye Hong, Xun Peng, Wenbin Li, and Eamonn O'Neill. 2022. Freehand Gestural Selection with Haptic Feedback in Wearable Optical See-Through Augmented Reality. *Information* 13, 12 (2022). doi:10.3390/info13120566

- [67] Xizi Wang, Ben Lafreniere, and Jian Zhao. 2024. Exploring Visualizations for Precisely Guiding Bare Hand Gestures in Virtual Reality. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) (CHI '24). Association for Computing Machinery, New York, NY, USA, Article 636, 19 pages. doi:10.1145/3613904.3642935
- [68] Colin Ware and Harutune H. Mikaelian. 1986. An evaluation of an eye tracker as a device for computer input2. *SIGCHI Bull.* 17, SI (May 1986), 183–188. doi:10.1145/30851.275627
- [69] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Proceedings of the 36th International Conference on Neural Information Processing Systems* (New Orleans, LA, USA) (NIPS '22). Curran Associates Inc., Red Hook, NY, USA, Article 1800, 14 pages.
- [70] Julia Woodward, Feben Alemu, Natalia E. López Adames, Lisa Anthony, Jason C. Yip, and Jaime Ruiz. 2022. “It Would Be Cool to Get Stamped by Dinosaurs”: Analyzing Children’s Conceptual Model of AR Headsets Through Co-Design. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) (CHI '22). Association for Computing Machinery, New York, NY, USA, Article 152, 13 pages. doi:10.1145/3491102.3501979
- [71] Liwei Wu, Ben Lafreniere, Tovi Grossman, Thomas White, and Stephanie Santosa. 2024. Body Language for VUIs: Exploring Gestures to Enhance Interactions with Voice User Interfaces. In *Proceedings of the 2024 ACM Designing Interactive Systems Conference* (Copenhagen, Denmark) (DIS '24). Association for Computing Machinery, New York, NY, USA, 133–150. doi:10.1145/3643834.3660691
- [72] Haijun Xia, Michael Glueck, Michelle Annett, Michael Wang, and Daniel Wigdor. 2022. Iteratively Designing Gesture Vocabularies: A Survey and Analysis of Best Practices in the HCI Literature. *ACM Trans. Comput.-Hum. Interact.* 29, 4, Article 37 (may 2022), 54 pages. doi:10.1145/3503537
- [73] Xin Zeng, Xiaoyu Wang, Tengxiang Zhang, Chun Yu, Shengdong Zhao, and Yiqiang Chen. 2024. GestureGPT: Toward Zero-Shot Free-Form Hand Gesture Understanding with Large Language Model Agents. *Proc. ACM Hum.-Comput. Interact.* 8, ISS, Article 545 (Oct. 2024), 38 pages. doi:10.1145/3698145
- [74] Shumin Zhai, Carlos Morimoto, and Steven Ihde. 1999. Manual and gaze input cascaded (MAGIC) pointing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Pittsburgh, Pennsylvania, USA) (CHI '99). Association for Computing Machinery, New York, NY, USA, 246–253. doi:10.1145/302979.303053
- [75] Yuchong Zhang, Adam Nowak, Yueming Xuan, Andrzej Romanowski, and Morten Fjeld. 2023. See or Hear? Exploring the Effect of Visual/Audio Hints and Gaze-assisted Instant Post-task Feedback for Visual Search Tasks in AR. In *2023 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. 1113–1122. doi:10.1109/ISMAR59233.2023.00128

A EXPERIMENTATION FOR FORMAL REPRESENTATION

Approach 1: Raw Coordinates Initially, we fed raw 3D fingertip coordinates from the OVR Skeleton into the LLM prompt (GPT-4o) to identify gesture’s pose, location and placement relative to virtual object. While coarse hand poses (e.g., finger pinching) were occasionally recognized, LLM struggled with finer details such as finger curls, subtle finger variations, and was generally unable to detect dynamic hand movements, highlighting the limitations of using raw positional data alone.

Approach 2: Introducing Temporal Differences on Raw Coordinates To capture dynamic trajectory of the hand pose over time, we enhanced the raw data with a 5-second time-differential, comparing hand skeleton data of the current frame with pose state from 5 seconds prior. We evaluated several time windows—3, 5, 7, and 10 seconds, and found that a 5-second window best captured the dynamic trajectories of the gestures for our initial feasibility study. This temporal context enabled the LLM to infer movement direction and relative pose changes, yielding moderate improvements in detecting location-based gestures (e.g., swipes). However, the pose accuracy (e.g., whether fingers were angled or curved correctly) remained limited because the representation still relied on raw coordinates.

B STUDY 1: INITIAL FEASIBILITY STUDY DETAILS

B.1 Participants

Our participants had an interest in using these technologies in the future. Only 4 participants had tried out the Meta (Oculus) Quest 2 headset (with controllers) before. Our participants represented a range of age groups: 18-24 (35%), 25-34 (58%), 35-44 (7%) and had different levels of education (6 Bachelor’s, 4 Master’s, 5 PhD). We recruited participants from our university’s mailing lists and snowball sampling.

B.2 Study Design

For both the studies, we used Meta Quest 3 AR head-mounted display, featuring 2064×2208 pixels per eye, a 4K+ Infinite Display with 25 pixels per degree (PPD) and 1218 pixels per inch (PPI), two full-color RGB cameras at 18 PPD for passthrough resolution, integrated stereo speakers with 3D spatial audio and external audio output, and a field of view of approximately 10° horizontally and 90° vertically.

B.3 Metrics and Ground Truth for the Feasibility Study

We measured ARGESTUREAID’s accuracy in detecting errors and providing LLM-driven corrective assistance across two phases: Gesture Discovery and Gesture Execution. Our metrics included:

B.3.1 Error detection Accuracy ($G_{Error-Accuracy}$). To determine whether a user’s gesture is correct, the ARGESTUREAID follows a predefined set of criteria known as the Optimum Gesture Reference Criteria (see Table 5). If the recognized user’s gesture is incorrect for the intended task, it is flagged as an error and is passed to the LLM for error recovery assistance. To evaluate error detection accuracy, we compared actual user gesture errors (identified from video recordings) with those detected by ARGESTUREAID through its system output logs. These logs represent ARGESTUREAID’s natural language interpretations of gesture errors, hand pose during discovery and gesture placement relative to virtual objects during execution, generated by the LLM-driven ERROR DETECTION Module (See Figure 1), which take formal hand gesture representations as input.

Gesture Discovery ($Error_{GD}$): We evaluated pose and location errors and our scoring criteria included:

- **Step 1 (Pose):** We assigned 0.2 points for each finger pose that ARGESTUREAID correctly detects, yielding a maximum of 1.0 point for accurate full-hand pose detection.
- **Step 2 (Orientation and/or Location):** Orientation accuracy of the gesture was assessed for dynamic pose gestures (e.g., rotating a box) where the gesture’s orientation changes over time. As shown in Table 5, an

Table 5. Optimum Gesture Reference Criteria outlining how ARGESTUREAID determines whether a gesture is correctly recognized (after 5 seconds) before passing it to the LLM for error recovery assistance. * Refers to bimanual gestures.

Gesture	Pose	Location
Static Pose-and-Path [Swiping Tiles]	$d_{im} < 1 \text{ cm} \wedge d_{mr} > 8 \text{ cm} \wedge d_{mp} > 8 \text{ cm}$	$\ \text{centroid}(t) - \text{centroid}(t - 5)\ > 10 \text{ cm}$ If the right hand's centroid moves more than 10 cm over 5 seconds, it is classified as a swipe location.
Dynamic Pose [Rotating a box]	$d_{it} < 1 \text{ cm} \wedge d_{im} > 8 \text{ cm} \wedge \Delta\theta > 30^\circ$	N/A
Dynamic Pose-and-Path [Grabbing a box]	$d_{it} < 1 \text{ cm} \wedge d_{im} > 8 \text{ cm}$	$\ \text{centroid}(t) - \text{centroid}(t - 5)\ > 20 \text{ cm}$ If the right hand's centroid moves more than 20 cm over 5 seconds, it is classified as grabbing.
Dynamic Pose-and-Path* [Zooming a rectangle]	$(d_{it}^{(L)} < 1 \text{ cm} \wedge d_{im}^{(L)} > 8 \text{ cm}) \wedge (d_{it}^{(R)} < 1 \text{ cm} \wedge d_{im}^{(R)} > 8 \text{ cm}) \wedge d_{\text{centroid}} > 20 \text{ cm}$	$\ \text{centroid}(t) - \text{centroid}(t - 5)\ > 20 \text{ cm}$ If each hand's centroid moves more than 20 cm over a 5 second interval, the gesture is classified as two-hand zoom.

ideal gesture rotation exceeds 30° . ARGESTUREAID received 1.0 point when it correctly detected orientation errors for rotations $\leq 30^\circ$, otherwise 0. Location accuracy of gestures was evaluated by measuring hand centroid displacement for static and dynamic pose-and-path gestures involving simultaneous changes in both hand pose and spatial position. For static pose-and-path gestures (See Table B.3), such as swiping tiles, ideal displacement exceeded 10 cm over 5s; for dynamic gestures (e.g., grabbing or two-hand zoom), it exceeded 20 cm. For bimanual gestures, the 20 cm threshold applied to each hand. ARGESTUREAID received 1.0 point for correctly detecting location errors (≤ 10 cm for static, ≤ 20 cm for dynamic), otherwise 0.

Gesture Execution ($Error_{GE}$): During execution, we measured the Euclidean distance between the fingertip and the virtual object's center. Gesture execution within 2 cm were considered correct; otherwise, errors were recorded. ARGESTUREAID received 1.0 point for correctly detecting an execution error, otherwise 0. For each user attempt i^{th} , discovery and execution error detection scores are denoted as d_i and e_i , with N_d and N_e representing the total user attempts during discovery and execution respectively:

$$\text{Avg. Error}_{GD} = \frac{\sum_{i=1}^n d_i}{N_d}; \quad \text{Avg. Error}_{GE} = \frac{\sum_{i=1}^n e_i}{N_e}$$

$$\text{Error Accuracy} = \frac{(\text{Avg. Error}_{GD} \times N_d) + (\text{Avg. Error}_{GE} \times N_e)}{N_d + N_e}$$

B.3.2 LLM-driven Adaptive Assistance Accuracy (Assistance Accuracy): We evaluated how accurately ARGESTUREAID provided adaptive corrective assistance in response to users' error and queries. We compared LLM's responses against the ideal response (explained in detail below) grounded on formal representation involving correct pose, and location during gesture discovery and correct gesture placement relative to the virtual object during gesture execution.

Assistance Accuracy for Gesture Discovery (Accuracy_{GD}): During the discovery phase, we evaluated ARGESTUREAID's guidance for correcting pose, orientation, and location errors. ARGESTUREAID was assigned 0.5 points for accurately specifying each error type detected (pose and orientation/location) and 0.5 for providing accurate adjustments in finger positions or orientation to correct each detected error (max 1.0).

Assistance Accuracy for Gesture Execution (Accuracy_{GE}): During execution, we evaluated the accuracy of ARGESTUREAID for executing the gesture relative to virtual objects. Correct execution required hand placement within 2 cm of the object center. ARGESTUREAID received 0.5 points for accurate feedback on error identified and 0.5 for precise corrective guidance (max 1.0 per attempt). For each user attempt i^{th} , discovery and execution assistance accuracy scores are denoted as d_{assist}^i and e_{assist}^i , with N_d and N_e representing total user attempts during discovery and execution.

$$\text{Avg. Accuracy}_{GD} = \frac{\sum_{i=1}^n d_{assist}^i}{N_d}; \quad \text{Avg. Accuracy}_{GE} = \frac{\sum_{i=1}^n e_{assist}^i}{N_e}$$

$$\text{Assistance Accuracy} = \frac{(\text{Avg. Accuracy}_{GD} \times N_d) + (\text{Avg. Accuracy}_{GE} \times N_e)}{N_d + N_e}$$

C PROMPT GIVEN TO LLM

C.1 An Example of Intent Detection Prompt

```
{"role": "system", "content": "Intent Detection: Your only task is to find
the user's intent."},
{"role": "user", "content": "User statement: [userSpeech] Last LLM response
: [lastUserLLMResponse]
```

1. If the user's intent is unclear, ask what they want to do and explain you can help with:
 - Rotate a box
 - Zoom in on a tile
 - Swipe tiles
 - Move a box
 - Create a circle / sphere / ball
 - Increase size of a box

Then ask which task the user wants to accomplish.

2. If the intent is clear, return a one-line interpretation mapping it to the closest supported task.
3. Prioritize making the best guess. Only ask for clarification when truly unable to infer intent.
4. If the user asks an off-topic question, answer briefly in one line.
5. Return either: 'not sure what you mean, could you please explain your task?' OR 'I understand, you would like to...' [one line interpretation]''}

C.2 An Example of Gesture and Error Detection Prompt

```

{"role": "system", "content": "Gesture Detection: Given the hand data below
, interpret the pose of both hands. Describe the position of each finger
, which are folded, which are extended, and which are close to or
touching each other. Provide output directly without any preamble."},
{"role": "user", "content":
"hand data format:
  distance between hands: float,
  left fingertip distances: [
    thumb-index, index-middle, middle-ring, ring-pinky,
    thumb-middle, thumb-ring, thumb-pinky
  ],
  right fingertip distances: [
    thumb-index, index-middle, middle-ring, ring-pinky,
    thumb-middle, thumb-ring, thumb-pinky
  ],
  rotation difference: float,
  distance from object: float,
  distance from screen: float

  hand data: [currData]}

```

C.3 An Example of Adaptive Assistance Prompt

```

{"role": "system", "content": "Coaching Prompt: Be professional and
supportive. If the user asks a question, answer it briefly first, then
continue coaching. Based on the user's current hand pose and the gesture
requirements in static instructions, identify the most important error
the user is making and guide them to correct it in one short friendly
line.

```

Follow this coaching order, address only one stage per response:

1. If the hand pose is incorrect, point out what is wrong and how to fix it.
2. If the pose is correct, guide the movement if the gesture requires one.
3. If pose and movement are correct, advise the user on positioning relative to the object.

Keep the response under 18 words. Avoid technical jargon, numbers, or internal system terms. Be natural and conversational."},

```

{"role": "user", "content":
"User said: '[userSpeech]'"
static instructions: [staticInstructions]

```

```

hand pose: [handPose]
previous LLM response: [lastLLMResponse]}
Example of staticInstructions:
Pinch and Rotate:
    Pose: Only right thumb and index finger pinched tightly together,
    with middle, ring, and pinky fingers open and extended.
    Movement: A noticeable clockwise wrist rotation is required.
    Proximity: Hand must be close to the object to interact with it.

```

D STUDY 2 DETAILS

Table 6. Overview of Study 2 Tasks.

Task Instructions Provided to the Participants	Gesture Form		Unimanual / Bimanual	Correct Gesture
You'll see a bunch of menu tiles (starting from yellow) in front of you. Your task is to move these tiles horizontally starting from yellow so that you can have green tile in front of you. Once you locate the green color tile, "select" it to change the color of the grey screen to green	Static Pose-and-Path		Unimanual	Two-finger swipe
Once you've created the sphere, your task is to "grab" the circle created and "move" it	Static Pose-and-Path		Bimanual	Circle gesture with both hands, fingertips touching and rotating this gesture clockwise to manipulate the sphere's orientation
You'll see a green box in front of you. You want to explore how this box will look if its size is increased in your 3D space. Your task is to increase the size of this box in front of you	Dynamic Pose		Unimanual	Thumbs-up gesture with lateral thumb movements to scale the box
You'll see a grey box placed on the table, your task is to "rotate" it clockwise to adjust its orientation upside down, positioning the top of the box as the bottom	Dynamic Pose		Unimanual	Pinch-Rotate gesture
Your task is to "zoom in" on the grey tile (a rectangular tile placed in the middle) to enlarge it from both sides	Dynamic	Pose-and-Path	Bimanual	Two-hand zoom and extend
Create a sphere in 3D	Dynamic	Pose-and-Path	Bimanual	Circle gesture with both hands, fingertips touching and then grab and move horizontally