

RAVEN: Realtime Accessibility in Virtual ENvironments for Blind and Low-Vision People

Xinyun Cao
Computer Science and Engineering
University of Michigan
Ann Arbor, Michigan, USA
xinyunc@umich.edu

Kexin Phyllis Ju
School of Information
University of Michigan
Ann Arbor, Michigan, USA
kexinju@umich.edu

Chenglin Li
Electrical and Computer Engineering
University of Michigan
Ann Arbor, Michigan, USA
lchengl@umich.edu

Venkatesh Potluri
School of Information
University of Michigan
Ann Arbor, Michigan, USA
potluriv@umich.edu

Dhruv Jain
Computer Science and Engineering
University of Michigan
Ann Arbor, Michigan, USA
profdj@umich.edu

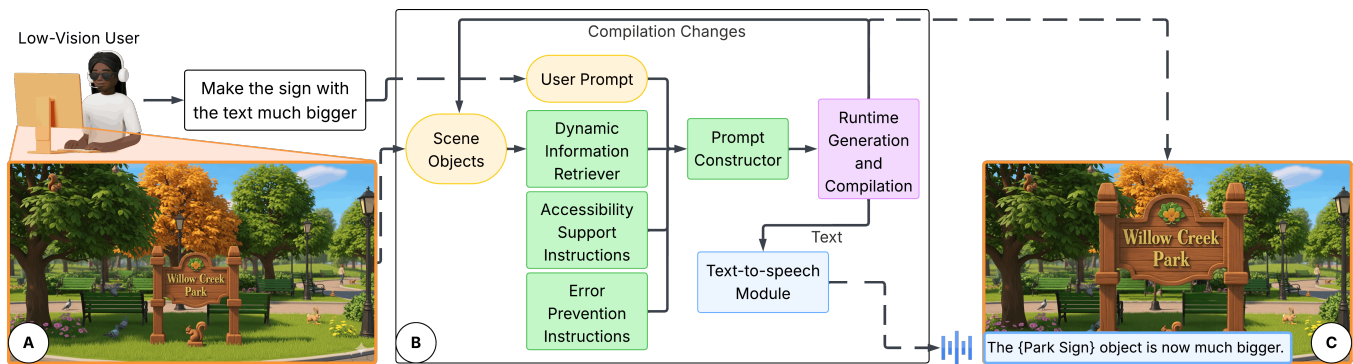


Figure 1: System workflow of RAVEN, an interactive tool that enables BLV users to adapt 3D scenes through conversational natural language. The figure illustrates an example interaction: (A) A low-vision user types, “Make the sign with the text much bigger.” (B) The system combines semantic scene data with a runtime code-generation LLM to translate this request into an accessibility-enhancing modification. (C) The generated code is compiled and executed in real time, while the system also provides a spoken response confirming the change.

Abstract

As virtual 3D environments become more prevalent, equitable access is essential for blind and low-vision (BLV) users, who face challenges with spatial awareness, navigation, and interaction. Prior work has explored supplementing visual information with auditory or haptic modalities, but these methods are static and offer limited support for dynamic, in-context adaptation. Recent advances in generative AI allow users to query and modify 3D scenes via natural language, introducing a paradigm that offers greater flexibility and control for accessibility. We present RAVEN, a system that enables BLV users to issue queries and modification prompts to improve the runtime accessibility of 3D virtual scenes. We evaluated RAVEN with eight BLV people and six Unity developers, generating empirical insights into how conversational programming can support

personalized accessibility in 3D environments. Our work highlights both the promise of natural language interaction—intuitive, flexible, and empowering—and the challenges of ensuring reliability, transparency, and trust in generative AI-driven accessibility systems.

CCS Concepts

• **Human-centered computing** → **Accessibility systems and tools.**

Keywords

Accessibility, 3D, blind and low-vision, generative AI



This work is licensed under a Creative Commons Attribution 4.0 International License. *CHI '26, Barcelona, Spain*

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

ACM ISBN 979-8-4007-2278-3/26/04

<https://doi.org/10.1145/3772318.3791616>

ACM Reference Format:

Xinyun Cao, Kexin Phyllis Ju, Chenglin Li, Venkatesh Potluri, and Dhruv Jain. 2026. RAVEN: Realtime Accessibility in Virtual ENvironments for Blind and Low-Vision People. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems (CHI '26)*, April 13–17, 2026, Barcelona, Spain. ACM, New York, NY, USA, 19 pages. <https://doi.org/10.1145/3772318.3791616>

1 Introduction

Virtual 3D environments have become pervasive, supporting a diverse range of applications such as interactive gaming, social networking, and education. These immersive spaces enable rich interactions, offering users a sense of presence and spatial exploration previously inaccessible in traditional media [12, 47]. However, with their increased adoption arises the imperative of ensuring inclusive and equitable access, particularly for blind and low-vision (BLV) users who encounter challenges in spatial understanding, navigation, and object interaction in these environments [84]. To address these challenges, prior work proposed tools that modify or supplement visual information using alternative modalities, such as audio descriptions [30], haptic feedback [38, 94], and enhanced visual effects tailored for low vision [95]. Systems like SceneWeaver [5] offer users greater agency over when and how to consume scene descriptions. Commercial platforms have also adopted accessibility features—such as high-contrast display modes and spatial audio—to accommodate BLV players in mainstream video games [7, 64]. These methods allow BLV users to utilize pre-defined tools to enhance the accessibility of a virtual 3D environment.

However, current approaches to accessibility share a fundamental limitation: they are predominantly developer-driven and static, such as fixed mappings for color changes or auditory overlays, and may not align with the nuanced and evolving needs of individual users [21]. Moreover, they often require users to learn specific control mappings or adjust settings in non-intuitive ways, leading to a steep learning curve while offering limited support for dynamic, context-specific adaptation.

Recent advances in generative AI (GenAI), particularly large language models (LLMs), open possibilities for more flexible and conversational interaction. LLMs have been increasingly applied to accessibility tasks such as querying images [1] and runtime scene editing [39], suggesting the potential for users to directly query and adapt 3D scenes through natural language, bypassing rigid developer-defined workflows and expanding user agency.

Motivated by this potential, we present RAVEN, an interactive system that empowers BLV users to engage with 3D scenes via natural language interaction. RAVEN supports both querying (e.g., “What’s around me?”) and accessibility-related modifications (e.g., “Make the table brighter” or “Move the bench closer”). It integrates LLMs with semantic scene data and runtime code generation to apply changes at runtime, while providing spoken responses to user queries. Interaction is iterative, enabling users to refine modifications through follow-up prompts in a dialogue-like flow.

We evaluated RAVEN with eight BLV participants in a user study including three interactive scenarios: a guided tutorial, structured tasks, and free exploration. The scenarios and study tasks were designed around six accessibility categories drawn from prior work. Participants engaged in conversational interactions to tailor scenes according to their accessibility preferences, offering quantitative feedback on usability and qualitative reflections on their experiences. Findings reveal critical insights into the potential and limitations of generative AI-supported interactions, highlighting both opportunities for enhanced accessibility and challenges with trust and reliability. We also conducted a preliminary study with

six Unity developers to evaluate the system’s usability from a developer perspective, yielding insights into its learnability, overall usability, and opportunities for improvement.

In summary, our work contributes: (1) the design and implementation of RAVEN, a GenAI-powered system for real-time, natural language-driven querying and modification of 3D environments for accessibility, and (2) empirical insights from a study with eight BLV users and a preliminary study with six Unity developers, highlighting interaction strategies, usability, and challenges that inform future LLM-based accessibility tools. Our work demonstrates the importance of flexible, intuitive, and contextually adaptive accessibility in virtual environments.

2 Related Work

We review prior work on accessibility in virtual 3D environments for BLV people, survey how generative AI has been used to enhance access across contexts, and examine runtime generative tools for virtual 3D scenes that inform our system’s capabilities.

2.1 BLV Accessibility in Virtual 3D Environments

For the context of this work, virtual 3D environments are computer-generated three-dimensional spaces that may be displayed on PCs, mobile devices, or immersive VR headsets. Early efforts created bespoke virtual environments for BLV users [4, 30, 63, 76], for activities including rehabilitation and orientation training programs [70, 77] and specific experiences such as boxing [25], racing [29], and table tennis [41]. While these systems demonstrate feasibility and value, their design assumptions often limit generalizability beyond the particular contexts for which they were built.

To broaden access in general-purpose 3D spaces, researchers have explored substituting visual information via audio and haptics. Guerreiro *et al.* articulate a design space for auditory substitution in virtual environments, providing a theoretical framework for cue design [33]. Commercial haptic hardware, however, typically lacks the spatial resolution to convey rich scene structure, so haptics is commonly paired with audio [5, 56] or requires custom devices [77, 89, 94]. A growing thread emphasizes agency and free exploration. Canetroller provides an auditory-haptic “white cane,” enabling transfer of real-world cane skills into VR for learnable exploration [94]. NavStick supports surveying surroundings through a gamepad thumbstick with auditory feedback, improving mental-map accuracy over menu-based baselines [56]. SceneWeaver gives users control over when and how to receive descriptions, increasing exploratory agency [5]. This body of work shows how auditory and haptic feedback can help BLV users understand and navigate 3D scenes.

For users with residual vision, enhancing the visual modality itself can be effective. SeeingVR demonstrates how magnification, contrast enhancement, recoloring, and related tools help low-vision users complete tasks more quickly and accurately [95].

Many of these ideas have influenced commercial games, which now include accessibility settings such as navigation assistance, combat audio cues, and high-contrast modes [7, 64], offering evidence of long-term learnability and adoption.

Despite these advances, most approaches remain developer-defined and static: substitutions (e.g., auditory cues) and modifications (e.g., color mappings) are predetermined and may not track the diverse, evolving needs of BLV users [21]. They can also impose learning burdens through mode switches, keybindings, or complex settings, which is especially challenging for newcomers [44, 51]. To our knowledge, RAVEN is the first system to enable BLV users to *query* and *modify* virtual 3D scenes *at runtime* via natural language, shifting control from developer-defined presets to user-directed, in-situ adaptation.

Despite these advances, most approaches remain developer-defined and static: substitutions (e.g., auditory cues) and modifications (e.g., color mappings) are predetermined and may not track the diverse, evolving needs of BLV users [21]. They can also impose learning burdens through mode switches, keybindings, or complex settings, which is especially challenging for newcomers [44, 51].

We also build on a prior non-archival ASSETS demo that presented an early prototype based on our work [15]. That demo introduced the idea of using LLM-driven runtime modifications in Unity and reported preliminary results, but it did not include the full analysis of the BLV user study or the developer study reported here. In this paper, we substantially extend that prototype by (1) fully specifying the system architecture and prompt-engineering framework (section 3.3.1-section 3.3.6), (2) deriving and documenting accessibility rules and categories from BLV literature, and (3) contributing two new empirical evaluations: a three-scene study with eight BLV participants and a preliminary usability study with six Unity developers. We therefore treat the demo as an early proof-of-concept and this paper as the first archival presentation of the complete system and its evaluation.

To our knowledge, RAVEN is the first system to enable BLV users to *query* and *modify* virtual 3D scenes *at runtime* via natural language, shifting control from developer-defined presets to user-directed, in-situ adaptation.

2.2 Generative AI for Visual Accessibility

Generative AI (GenAI) tools have increasingly supported BLV users' everyday access needs, including real-world scene understanding [8, 17], navigating digital interfaces [42], visual authoring [18], and professional productivity [62, 71]. These systems leverage natural language as an intuitive control channel and adapt responses to context and user goals. For example, Savant allows users to control screen readers conversationally, reducing dependence on complex shortcut vocabularies and lowering workload [42]. WorldScribe provides real-time, context-aware descriptions, offering concise updates for dynamic scenes and more detailed accounts for stable ones [17]. This line of work motivates us to introduce similarly flexible, conversational interaction to virtual 3D contexts.

GenAI also enables visual content creation, spanning images [11, 93] and 3D scenes [36, 90], opening avenues for creative agency among BLV users [10]. Yet generated content is often inaccessible to its creators, complicating evaluation, verification, and iterative refinement [37]. To address this in 2D settings, GenAssist combines vision-language and language models to pose verification and style questions, and synthesize question-guided descriptions that help

users assess prompt alignment and compare candidates [37]. AltCanvas introduces a spatially structured, tile-based interface to improve layout comprehension while adding, editing, and moving visual components [45]. While effective in 2D, analogous accessibility support for *virtual 3D* generative workflows remains under-explored, where BLV users face additional demands for spatial and embodied understanding.

Adoption of GenAI in accessibility also introduces risks. A recent study by Glazko *et al.* [28] found that LLMs could recite accessibility guidance but struggled to apply it in context [28]; an empirical audit found high rates of accessibility issues in generated web code, with many defects persisting even after remediation prompts [3]. Despite these errors, the apparent fluency of agents can foster false expectations, overconfidence, and inaccurate mental models [1, 28]. The potential for hallucination further necessitates trustworthy verification and avenues for user contestation [1, 2, 91]. Finally, training data can encode biases and ableist stereotypes [26–28, 50, 79]. RAVEN attempts to address these concerns by embedding specific accessibility instructions into prompts for the Unity3D environment, lowering sampling temperature to reduce randomness [81], and explicitly detecting “out-of-scope” user requests to avoid spurious outputs.

Overall, RAVEN leverages LLM-generated responses and code to support accessibility in virtual 3D environments, aiming to combine the flexibility of conversational interfaces with safeguards that mitigate known GenAI limitations.

2.3 Runtime Generation in Virtual 3D Scenes Using GenAI

Recent work demonstrates GenAI systems for 3D scene *creation* [36, 87, 90], *modification* [19, 22, 39], and *interaction* [35, 46]. Collectively, these systems show that models can reason about spatial layouts, process multi-modal inputs, and generate executable changes-capabilities that are promising for accessibility.

Creation-focused systems primarily support 3D designers. SceneCraft generates Blender-executable Python scripts from language, rendering assets programmatically [36]. HoloDeck uses GPT-4 to select and place objects from model collections to instantiate embodied AI environments [87]. VRCopilot supports multi-modal furniture layout design to enhance immersion and creativity [90]. These systems illustrate how GenAI can interpret spatial intent and produce code to realize 3D content.

Beyond creation, GenAI can edit scenes and generate behaviors at runtime. GROMIT encodes Unity scenes as semantic graphs to enable LLM-authored behaviors and was evaluated with developers creating new game mechanics [39]. LLMR provides a framework for real-time creation and modification of mixed-reality experiences [22]. Chen *et al.* analyze user input patterns for LLM-assisted manipulation in immersive settings, identifying design considerations around user agency and handling uncertainty and hallucination [19]. This line of work suggests rich opportunities for accessibility-oriented modifications.

Other systems employ LLMs to mediate interaction in 3D spaces. HandProxy lets users command a virtual hand via natural language to operate UI and manipulate objects [46]. GesPrompt augments

this paradigm with co-speech gesture for lower-effort, more intuitive control [35]. These approaches echo non-GenAI proxies such as VoiceAttack [82] but also leverage multi-modal interaction for greater expressiveness and ease of use.

Despite these advances, utilizing runtime generation for *accessibility* remains underexplored. While systems such as LLMR hint at adaptations for color blindness, near-sightedness, or child-friendly design [22], they do not address the breadth of BLV needs nor evaluate with BLV users. Our work is the first to investigate LLM-driven, runtime scene querying and modification for accessibility with and for the BLV community.

3 Iterative System Design

RAVEN supports real-time queries and modifications in virtual 3D environments to improve accessibility for BLV users. The system is designed to be open, where users can use any natural language prompt. However, to ground its utility in user needs, we first developed an initial prototype and conducted a pilot study with three BLV participants. Insights from this study informed iterative refinements, resulting in the final system presented here. In this section, we introduce an illustrative use scenario (section 3.1), describe the pilot study and resulting design modifications (section 3.2), and detail the final system architecture (section 3.3).

3.1 Use Scenario

Consider Dez, a low-vision gamer who sometimes uses screen readers but primarily relies on residual vision augmented with accessibility modifications. While playing a 3D game enhanced by RAVEN, Dez first requests a description of the scene. Within seconds, the system replies: “*You are located at a dark street corner... In front of you, there is a mysterious parked car... Further in the distance, two characters are talking on their phones...*”

Dez then asks the system to brighten the scene. In response, the system adds point light sources, which Dez further refines by requesting they be placed near the mysterious car. Curious, he follows up: “*How many lights are near the car now?*”, and the system provides a count. After iteratively adjusting brightness levels, Dez continues customizing the environment—highlighting the car in bright yellow, raising the pitch and volume of a key character’s dialogue, and enlarging the license plate text.

This scenario illustrates how users can iteratively adapt scene elements through natural language, tailoring the environment to their accessibility needs. Figure 2 highlights additional examples, showing that RAVEN can support a wide range of modifications.

3.2 Pilot Study and Iterative Modifications

We built an initial prototype of RAVEN based on the intended use scenario and insights from prior BLV accessibility research [54, 56, 84]. The prototype included keyboard shortcuts for selecting and bookmarking objects as well as a natural language interface for querying and modifying the environment. To evaluate its feasibility and identify areas for refinement, we conducted a pilot study with three BLV participants (two with residual vision and one blind). Participants explored two scenes designed to present distinct accessibility challenges: (1) a static scene with complex spatial relationships to probe exploration, and (2) a noisy scene with

overlapping conversations to simulate dynamic social interactions. After receiving training in system interactions, participants freely explored the scenes. We collected usability ratings, observational data, and semi-structured interview feedback, which informed the iterative modifications described below.

3.2.1 Removing Keyboard Shortcuts for Conversation-Only Interaction. In the initial prototype, participants could interact with the system in two ways: through keyboard shortcuts (e.g., selecting and bookmarking objects) and through natural language commands for querying and modification. Usability ratings indicated a mixed experience, with an average System Usability Scale (SUS) score of 75.83—suggesting reasonable usability but leaving room for improvement. Participants reported that remembering keyboard shortcuts created “*initial frustrations*” and described them as non-intuitive. Consistently, the selection and bookmarking features tied to keyboard commands received the lowest ratings. In contrast, natural language interaction was rated most highly and described as more intuitive. To reduce learning burden and cognitive load, we removed keyboard-based features and focused exclusively on conversational interaction.

3.2.2 Adding Egocentric Spatial Descriptions. In the initial prototype, the system described object locations using raw 3D coordinates (x, y, z). Participants reported that this format was unintuitive and difficult to map onto their own perspective in the scene. For example, one participant remarked, “*I don’t quite understand the coordinates,*” while another noted that the system failed to recognize egocentric references such as “*what’s in front of me.*” To address this limitation, we augmented object metadata with embodied spatial relations (*in front of, to the left/right, behind*) and relative distance from the player. We then instructed the LLM to use this metadata to generate egocentric, perspective-aware descriptions that aligned with how participants naturally referenced space.

3.2.3 Mitigating Errors and Hallucination. During the pilot, participants observed typical LLM shortcomings, including hallucination, randomness, and failure to handle vague requests gracefully. At times the system claimed to have executed a modification when none had occurred, or produced erroneous changes. To mitigate these issues, we added prompt-engineering strategies instructing the system to: (1) ask clarifying questions when requests are vague, (2) acknowledge and recover from errors reported by users, and (3) recognize out-of-scope requests (e.g., adding magnifiers) rather than attempting unsupported modifications. Although hallucination remains a limitation of generative AI technologies [59], these strategies improved the system’s reliability and transparency.

3.2.4 Summary of Iterative Refinements. Through iteration, we streamlined RAVEN toward conversational interaction, refined spatial reasoning with embodied relations, and improved robustness against hallucination. Participants navigate scenes using standard keyboard controls (arrow keys for movement; W,A,S,D for camera panning), while all accessibility queries and modifications are handled through natural language. This design reduces learning overhead while preserving the flexibility and power of real-time modifications.

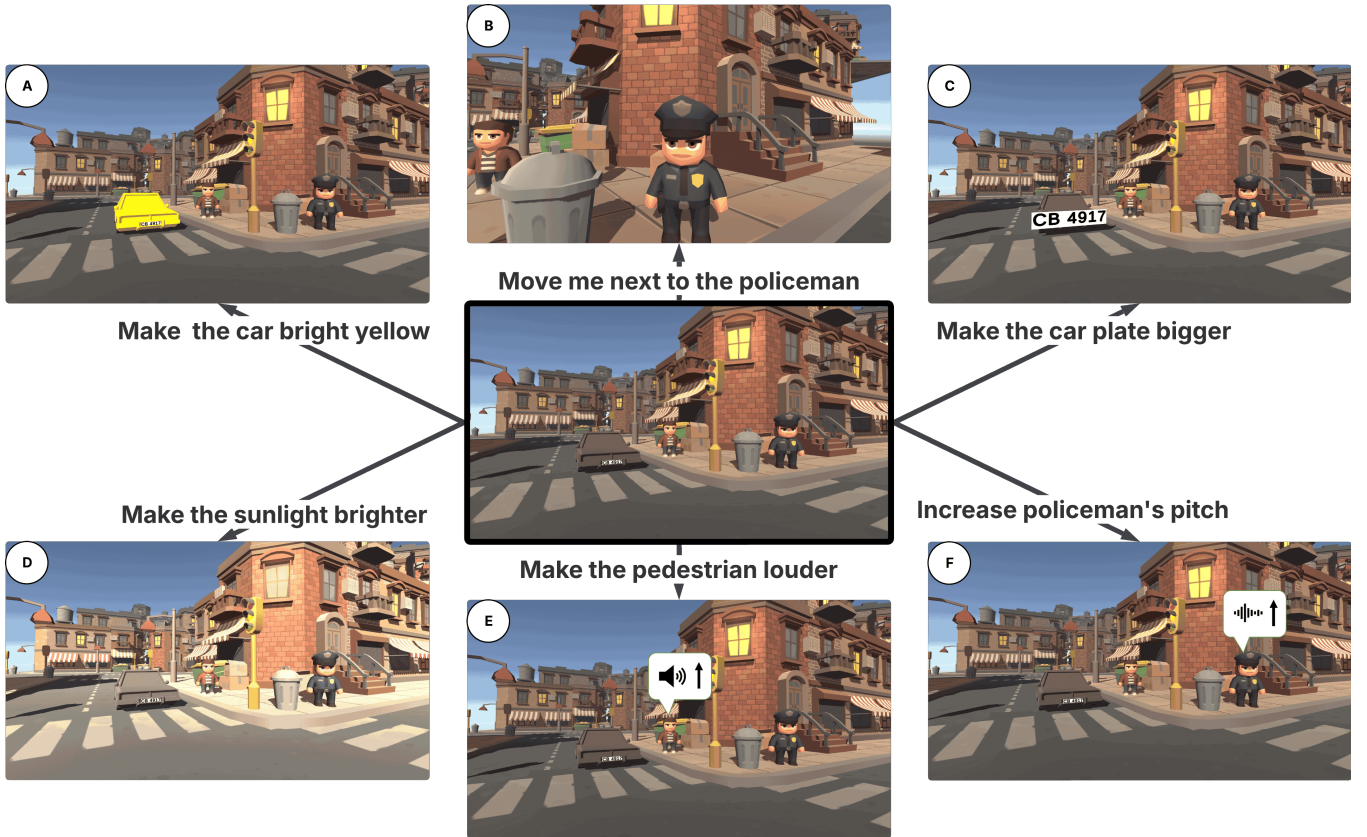


Figure 2: Examples of system usage. The center image shows the original game scene, with six surrounding panels illustrating accessibility-driven modifications: (A) change object color, (B) reposition the player, (C) enlarge a text object, (D) increase brightness, (E) amplify audio volume, and (F) adjust audio pitch. Bubbles and icons in E and F visualize auditory changes. These are only a few of the many types of modifications RAVEN can support to enable flexible, user-driven accessibility.

3.3 The RAVEN System

The final RAVEN architecture (fig. 1-B) requires only minimal developer annotation during scene creation (section 3.3.1) and integrates several components to support natural language interaction. At runtime, the system self-voices user input (section 3.3.2), retrieves an accessibility-augmented semantic scene graph (SSG) from the environment (section 3.3.3), and constructs prompts that combine user queries with accessibility and error-prevention instructions (section 3.3.4). These prompts are executed through the GROMIT runtime generation system [39], which generates Unity code to implement requested modifications and returns a textual response. This response is announced to the user through text-to-speech, completing the interaction loop. The full system implementation is open-sourced and can be accessed here: <https://github.com/SoundabilityLab/RAVEN>.

3.3.1 Scene Construction. To prepare a scene, developers create a 3D environment in Unity following standard workflows, then identify a subset of important objects (e.g., game items on a table). For each selected object, developers attach a lightweight Unity script and indicate whether it is physical (e.g., a table) or non-physical (e.g., an ambient sound source), and whether it is the player. Optional

developer-provided descriptions of visual or auditory properties may also be included. Developers also include lightweight scripts that connect the scene to the LLM agent. This metadata is the only additional input required, keeping developer burden minimal.

3.3.2 Text-to-Speech for Self-Voicing. Standard screen readers do not integrate well with Unity environments, yet audio feedback is critical for BLV users. To address this, RAVEN is designed as a self-voicing application: it announces characters as they are typed, reads words upon completion, and vocalizes system responses. We implemented this feature by building a wrapper around the Microsoft Speech API for Unity [53].

3.3.3 Dynamic Information Retriever. The Dynamic Information Retriever produces an up-to-date semantic scene graph (SSG) each time a prompt is issued, ensuring that the LLM receives current environmental context. Building on the structure introduced by Jennings *et al.* [39], we extend the SSG with accessibility metadata. For each relevant object, the SSG stores its name, developer-provided description (visual, auditory, and functional properties), attached scripts, position, scale, and hierarchical relationships. To support accessibility queries and modifications, the following metadata is recomputed at every prompt:

- (1) Color (HEX code).
- (2) Text content and font size.
- (3) Egocentric direction (*in front of, to the left/right, behind*) and distance from the player.
- (4) Light source density.
- (5) Audio source status (mute/unmute, volume, pitch, and range).

This enriched SSG provides the LLM with the contextual ground necessary for accurate descriptions and modifications.

3.3.4 Prompt Constructor. The Prompt Constructor fuses user input with scene data and prompt-engineered instructions before sending requests to the LLM. Prior research shows that LLMs often struggle with accessibility alignment and error handling [28, 40, 72]. To address this, the Prompt Constructor integrates two sets of instructions, which are included with every request as part of the conversational history.

1. Accessibility Support Instructions. Although LLMs may possess accessibility knowledge, they may struggle to apply it effectively in specific contexts [28]. We therefore synthesized accessibility needs in virtual 3D environments from prior BLV accessibility research [33, 56, 84, 95]. These needs span color/contrast adjustments, brightness changes, spatial understanding, text enlargement, scene description, and audio manipulation. We operationalized these needs into a set of explicit rules (e.g., how to adjust Unity C# variables, how to simplify materials before recoloring, and how to convert Unity terminology into user-facing language) that guide the LLM’s descriptions and modifications.

2. Error Prevention Instructions. To mitigate hallucinations and improve transparency, we designed instructions that enable the system to handle ambiguity and errors conversationally. If a request is incomplete or vague, the LLM asks clarifying questions (e.g., “*It seems like your request is not clear. Could you please provide more details or clarify what you would like to achieve?*”). If a user reports an error (e.g., “*it’s not working*”), the system is instructed to acknowledge the limitation and suggest alternative approaches. Finally, the LLM is primed [74] with a list of out-of-scope tasks identified during the pilot study (e.g., adding magnifiers), ensuring that unsupported requests are clearly surfaced rather than producing erroneous code.

Together, these measures ground the LLM in accessibility context, reduce hallucinations, and preserve the flexibility of conversational interaction. The full prompt is provided in appendix A.1.

3.3.5 Runtime Modification Generation and Compilation. Once the prompt is constructed, it is sent to GROMIT [39, 57], an open-source runtime behavior generation system for Unity. GROMIT processes the prompt by generating Unity code that implements the requested modification, attaches the compiled script to the relevant object, and returns a textual response. This enables the system to dynamically update scenes while providing users with immediate, self-voiced feedback. We used GPT-4o for language and code generation, as it was the most advanced model that supported feasible real-time integration during system development.

3.3.6 System Design Summary. In sum, RAVEN introduces a new approach to making virtual 3D environments accessible: BLV users

can issue natural language queries and modifications that are executed in real time. This is enabled by three key contributions: (1) a self-voicing interface that supports prompt entry and feedback, (2) an accessibility-augmented semantic scene graph that encodes perspective-aware relations and multimodal attributes, and (3) prompt-engineering strategies that align the LLM with accessibility goals and mitigate hallucinations. Together, these components shift control from static, developer-defined accommodations to dynamic, user-directed accessibility.

4 User Study Method

To evaluate the utility of on-demand accessibility modifications and the usability of RAVEN, we conducted a user study with BLV participants. The study focused on six accessibility categories (section 4.1) and three scenarios with progressively open-ended tasks (section 4.2). We addressed two research questions:

- **RQ1:** Can RAVEN support blind and low-vision people in experiencing a 3D scene?
- **RQ2:** What kinds of prompts do BLV participants use when interacting with the system to improve accessibility?

4.1 Accessibility Categories

To scaffold our study design, we synthesized six accessibility categories from prior work across game accessibility toolkits and empirical studies of BLV user needs [66, 67, 75, 80, 85, 95]. These categories were used solely to structure our evaluation tasks and analysis; they do not constrain RAVEN’s capabilities nor restrict what users may request during interaction. Four categories address the visual domain and two target the auditory dimension.

4.1.1 Visual Categories. These categories allow users to adapt the visual presentation of a scene:

- (1) **Color:** Retrieve or modify color schemes and object colors to aid recognition, especially for color-blind and low-vision users [66, 85].
- (2) **Object Location:** Retrieve or reposition objects (e.g., furniture, characters) to simplify navigation and interaction, inspired by assistive game toolkits [82].
- (3) **Object Size:** Resize objects or text to enhance visibility and readability [67, 80].
- (4) **Scene Brightness:** Query or adjust overall brightness or specific light sources to accommodate visual sensitivity [95].

4.1.2 Auditory Categories. These categories adapt the auditory experience of a scene:

- (1) **Audio Volume:** Increase or decrease the volume of sound sources to isolate or emphasize elements [16, 68].
- (2) **Audio Pitch:** Alter pitch to distinguish sounds, highlight important content, or emulate screen reader cues [16, 32, 58, 83].

4.2 Scenes

We designed three scenes to scaffold participants’ exploration of the six accessibility categories, with increasing complexity and freedom of interaction (fig. 3).

4.2.1 Scene 1: Guided Tutorial. The first scene introduced participants to the system in a simple, game-like room containing two geometric objects with distinct colors, two speakers playing dialogue, a torch with continuous sound and animation, a text object, and a table (fig. 3a). The researcher demonstrated each category using a three-step prompt pattern (ask, modify, verify), after which participants practiced similar prompts themselves. This structure, inspired by prior work on generative AI accessibility tools [1, 28, 37], ensured that participants could explore both querying and modification. Full list of demo prompts used are included in appendix A.2.

4.2.2 Scene 2: Task-Driven Exploration. The second scene, a cat park with natural elements, three cats producing distinct meows, a bench, a streetlamp, and a garden hut (fig. 3b), was paired with six predefined tasks (table 1). The tasks were designed to align with the six accessibility categories. After completing a task, participants reflected on their experience, including whether the interaction was intuitive, useful, or confusing. This setup allowed us to observe how participants applied categories in a goal-oriented context.

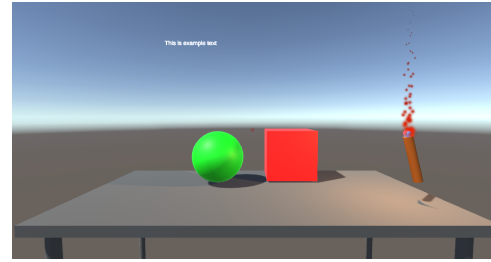
4.2.3 Scene 3: Open-Ended Exploration. The final scene encouraged open-ended use of the system. Participants explored a spaceship-themed room containing sixteen objects and three sound sources (fig. 3c). They were given ten minutes to query, modify, and adapt the environment according to their own needs, without predefined tasks. This scenario revealed how participants appropriated the system in less structured contexts.

4.3 Study Design

4.3.1 Participants. We recruited eight BLV participants (five men, three women; ages 24–50, $M=36.6$, $SD=7.9$) through mailing lists and word of mouth. Two participants were blind with no vision, two had light perception, and four had low vision. All used screen readers; some also used braille displays, large text, or commercial interpreting services (e.g., Be My Eyes, Aira). Three regularly used conversational AI apps (e.g., ChatGPT, Claude), four had limited experience, and one had never used them. Three had played games prior to vision loss, while five continued gaming using accessibility features. Full demographics are shown in table 2.

4.3.2 Study Setup. The scenes in the study were deployed in a 3D environment running on a laptop. Prior work shows that this setup could be effectively used to evaluate visual accessibility in 3D and 360 environments [23, 56]. We excluded more novel formats (e.g., VR) to keep the study centered on 3D spatial aspect and to avoid confounding the results with extra learning demands. Participants used a laptop device running Unity, a full-sized keyboard, and stereo headphones. The researcher also used stereo headphones to monitor system audio output in real time. See fig. 4.

4.3.3 Procedure. Studies lasted approximately 1.5 hours and were conducted either in-lab ($N=6$) or at participants' homes ($N=2$). Each session followed the three scene explorations (section 4.2), then a survey (usefulness of categories, confidence, intuitiveness, and SUS [14]), and finally a semi-structured interview (usability, new category ideas, prompt strategies, accessibility needs, AI trust). Sessions were audio- and video-recorded. Participants received a \$50 gift card and travel reimbursement. The study was IRB-approved.



(a) Scene 1



(b) Scene 2



(c) Scene 3

Figure 3: Screenshots of the three scenes used in our evaluation. Scene 1: a demo with simple objects and sound sources. Scene 2: a park with cats meowing and background nature sounds. Scene 3: a spaceship room with furniture, small items, and sci-fi sound sources.

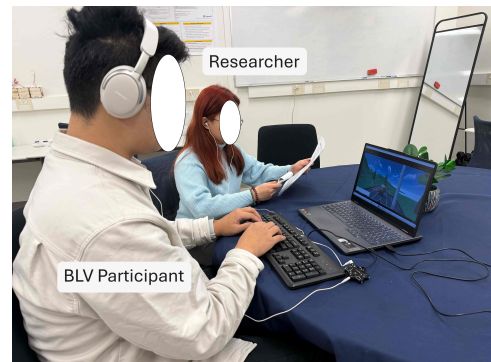


Figure 4: The study setup for the evaluation of RAVEN with BLV participants.

Table 1: Tasks in Scene 2

Number	Task
1	Ask the system to describe the scene.
2	The bench is a color unfriendly to low-vision users, can you change it to a better color?
3	Notice that there are some cats in the scene making some sounds. What’s different about how each of the cats sounds? Which of these cats seems to be the happiest?
4	Benches in the park are sometimes dedicated to people who donated money for them, and they have their name written in a short text on the benches. Where is the bench and who is this bench dedicated to?
5	Imagine you’re taking a (virtual) photo of the white cat, try increasing the brightness of the scene for a better photo.
6	Make the bench bigger so that you and the cat have space to sit together.

Table 2: Participant demographics. Visual ability labels: B = blind, LP = blind with light perception, VI = visually impaired (all self-identified). “Onset” indicates the age at which vision loss began. “AT usage” lists accessibility technologies used. “LLM usage” refers to experience with conversational AI apps.

Code	Age	Gender	Visual Ability	Onset	AT usage	LLM usage	Video game usage
P1-LP	34	Man	Light perception only	26	Screen readers	Never used	Played a few
P2-VI	43	Woman	Blurry and muted vision	Birth	Screen readers, braille display, and AI apps	Tried a few times	Played a few as kid
P3-VI	40	Man	Severe tunnel vision	Birth	Screen readers	Used regularly	Played a lot as kid, only haptic games recently
P4-LP	41	Man	Blind with light perception	Birth	Screen readers, braille display, and AI apps	Used regularly	Played a few
P5-VI	28	Man	Astigmatism in right eye	5	Large text, screen readers	Tried a few times	Played a few
P6-VI	24	Woman	Legally blind with Retinopathy of Prematurity	Birth	Large text, screen readers	Tried a few times	Played a few as kid
P7-B	33	Man	Totally blind with no light perception	Birth	Screen readers, braille display	Tried a few times	Play audio games regularly
P8-B	50	Woman	Totally blind with no usable vision	5	Screen readers, braille display, and AI apps	Used regularly	Never since vision loss

4.3.4 Data Analysis. We analyzed three sources of data: survey ratings, prompt logs, and interview transcripts. Together, these provided complementary perspectives on participants’ experiences and interaction strategies. The research team is mixed-ability, with one BLV, one DHH, and three able-bodied researchers. We acknowledge that these backgrounds shape our interpretation.

Survey Data. The survey was designed to address RQ1, evaluating RAVEN’s support for BLV users in 3D scenes. It included an SUS questionnaire [14] to evaluate system usability, and 5-point Likert scale questions about: usefulness of categories (section 4.1), user confidence in accessibility improvement, and intuitiveness of the system. We computed descriptive statistics (means, standard deviations) to summarize the data.

Prompt Logs. Prompts and system responses from Scenes 2 and 3 were recorded, yielding 336 valid prompts (181 from Scene 2 and 155 from Scene 3), after excluding the tutorial scene and erroneous inputs. We collected response time from recordings, measuring from the frame of user input submission to the frame when the

LLM finished loading and reply appeared. Each prompt was independently coded by multiple researchers along three dimensions, with disagreements resolved through discussion:

- (1) *Correctness*: whether the system response successfully achieved the request (success), correctly identified the request as out-of-scope, misaligned with user intent (intent error), or failed against ground truth (technical error).
- (2) *Command category*: the type of modification or query. Six predefined categories guided the study design, while analysis revealed four emergent categories, producing ten in total. Nine prompts were coded as “Other,” and eight combined multiple categories. We also calculated correctness rates per category.
- (3) *User goal*: the underlying intent behind the prompt (e.g., creative modification, verification). Through iterative coding, we developed three code groups encompassing eight codes. Prompts could receive multiple goal codes to capture complex intentions.

Interview Data. Interviews were transcribed and analyzed using applied thematic analysis [34]. Three researchers collaboratively: (1) familiarized themselves with the transcripts, (2) tagged data with codes, (3) developed an initial codebook, (4) refined codes and themes, (5) finalized the codebook, and (6) synthesized themes into findings.

This process produced a final codebook with 114 codes, organized into 30 third-level themes, 12 second-level themes, and 4 first-level themes. Before integrating qualitative insights into the Findings section, we summarize the structure of our thematic analysis. Our final codebook contained four major themes and twelve subthemes: (1) **System Performance and Reliability** (perceived accuracy and trust, handling of hallucinations, and verification strategies); (2) **Interaction and Feedback Experience** (intuitiveness of natural language interaction, appropriation of iterative prompting, and modality preferences across audio and visual cues); (3) **Prompt Categories and Use Patterns** (task-driven use in Scene 2, open-ended exploration in Scene 3, and differences between effective and ineffective prompting strategies); and (4) **Ethical and Broader Impacts** (concerns about safety and scene integrity, desires for guardrails and transparency, and reflections on broader accessibility opportunities and risks). These themes structure the qualitative findings reported in section 5 and clarify how we organized perspectives on system performance, interaction experience, prompting behavior, and broader impacts.

5 User Study Findings

Our analysis highlights both the promise and current limitations of RAVEN. We first report the system’s performance, followed by participants’ overall perceptions of usability, then examine performance across the six *guiding* and four *emerging* prompt categories, and finally describe the strategies and goals that shaped how participants engaged with the system.

5.1 System Performance Evaluation

Of the 336 valid prompts from Scenes 2 and 3 in the user study, 253 (75.3%) produced a correct query answer or intended modification (determined through researcher coding of session recordings), 9 (2.7%) were correctly flagged as out-of-scope, and 74 (22.0%) failed. Failures included *intent errors* (14 prompts, where the LLM misunderstood participant goals) and *technical errors* (60 prompts, where the LLM hallucinated, misrepresented system capabilities, or failed in code execution).

The average response time was 3.1 seconds ($SD=3.7$) across all prompts. A breakdown of the average response time for different correctness labels is shown in table 3. *Acknowledge Out-of-scope* prompts had the shortest average response time and the lowest variability. The remaining correctness labels had similar average response times, with *Technical Errors* requiring slightly longer.

5.2 Overall System Usability

Participants rated the system positively across confidence, intuitiveness, and usability. On average, they reported confidence in using the tool to make scenes accessible ($M = 4.1$, $SD = 0.8$, scale 1–5, 5 being best) and rated the system as intuitive ($M = 4.3$, $SD = 0.9$). The mean SUS score was 79.7, corresponding to an A– on the

Sauro–Lewis curved grading scale [69] and categorized as “good” usability [6].

Subjective feedback echoed these results. P1-LP observed that the tool “*seems very good at allowing someone to get a sense of the sort of overall environment they’re in and the sort of properties of items within that environment.*” P5-VI appreciated that adjustments could be made in-scene without interrupting gameplay, and P7-B compared the learnability favorably against audio games that require memorizing shortcuts: “*you could tell the AI to change something that you forgot the keyboard command for and then it could help you.*” Several participants ($N=2$) highlighted robustness to typos, ambiguity, and compound requests. Others ($N=2$) praised the open-endedness, with P8-B noting: “*the sky was the limit in some of the things that I could ask [the system] to do.*”

Five participants felt the system could enhance the gaming experience for BLV players. P6-VI described how it expanded her interest: “*[There were] games that I previously weren’t super interested in because of all the barriers related to the visual barriers that I face. [Having this tool] could potentially increase my interest in computer games.*” Participants also envisioned applications beyond games, including web design (P5-VI) and educational software (P4-LP), and for broader groups such as players with cognitive disabilities (P1-LP) or colorblindness (P7-B).

Despite these advantages, several limitations emerged. P4-LP described the gaps in flexible language and cross-modal consistency: extinguishing a torch stopped the light but left the fire sound. P6-VI similarly noted misapplied assumptions, such as turning all objects white after asking to “*make the room white*”. Participants ($N=7$) raised trust concerns about misleading responses. As P2-VI asked, “*If it gives the wrong information, how is a visually-impaired person going to know?*” P4-LP concluded: “*right now, [the system] is not so foolproof as to say I completely trust it.*” These findings highlight both the promise and current fragility of the approach. Because our participants’ visual abilities spanned a continuum—from no usable vision to varying forms of low vision—these impressions reflect a spectrum of experiences rather than a clean blind/low-vision split. This motivates our choice to report visual-ability differences qualitatively instead of conducting binary group comparisons.

5.3 Performance and Usage Across Prompt Categories

Qualitative coding of 336 prompts revealed ten categories: six *guiding categories* defined in study design (Object Location, Audio Volume, Color, Object Size, Scene Brightness, Audio Pitch) and four *emerging categories* that surfaced during analysis (Scene Description, Semantic Description, Functionality, Creation/Deletion). Nine prompts were coded as “Other” and eight combined multiple categories (e.g., “*Make the bench normal-sized (Object Size) and put us on the bench*” (Object Location)). See table 4 for code definitions and examples.

Figure 5 summarizes category occurrence and correctness. Object Location was most frequent, but with a relatively high error rate. Color and Audio Volume were also common, with high success rates. Semantic Description appeared often but was limited by out-of-scope acknowledgments. Functionality and Creation/Deletion were rare and error-prone, reflecting unsupported user needs.

Table 3: Counts and percentages of prompts per correctness label, and their corresponding average response times.

Correctness Label	Count	Percentage (%)	Avg. Response Time (sec)	SD of Response Time (sec)
Success	253	75.3	3.1	4.1
Acknowledge Out-of-scope	9	2.7	2.1	0.5
Intent Errors	14	4.2	3.2	1.6
Technical Errors	60	17.9	3.4	2.5
Total Errors (Intent and Technical)	74	22.0	3.2	2.3
All Prompts	336	-	3.1	3.7

Table 4: Prompt category codes with descriptions and examples.

Code	Description	Example
Object Location	Modify or describe the location of object(s) or the player.	“Move me closer to the white cat.”, “Where is the gold key?”
Audio Volume	Modify or describe the loudness of sound sources.	“Mute the laptop.”, “How loud is each cat?”
Color	Modify or describe the color of object(s).	“Make the bench bright yellow.”, “What color are the keys?”
Object Size	Modify or describe the size of object(s) or text.	“Increase the length of the bench.”, “How big is the laptop?”
Scene Brightness	Modify or describe brightness of lights or the scene.	“Increase skylight intensity.”, “How bright are the lamps?”
Audio Pitch	Modify or describe pitch of sound sources.	“Lower pitch of laptop to 0.4.”, “Which cat has the high-pitched meow?”
Scene Description	Describe the general scene or search for object(s).	“Describe the scene around me.”, “Does the room have a window?”
Semantic Description	Describe objects based on semantic qualities.	“Which cat seems happiest?”, “What is on the laptop screen?”
Functionality	Use objects functionally.	“Open the laptop.”, “Answer the phone.”
Creation/Deletion	Add or remove objects.	“Add a canopy to the bench.”, “Make the pen disappear.”

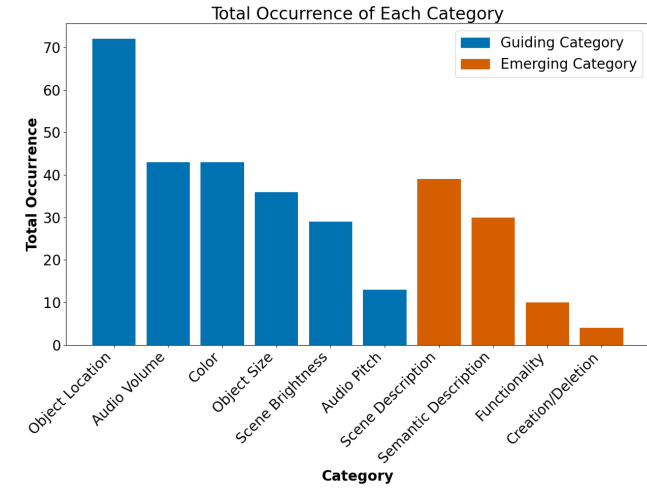
5.3.1 Guiding Categories.

Object Location. Most frequently used ($M = 4.8, SD = 0.7$ usefulness), this category enhanced spatial awareness through directional cues and repositioning. P4-LP stressed: “*You have to know how to orient in the world, and you have to be able to have it clear in space.*” Despite high value, success rates were lower than average due to LLM hallucinations (e.g., misaligned perspectives) and the system’s reliance on object-center coordinates, which provided only coarse control. This worked for broad moves (e.g., placing the player in a room) but broke down for finer actions (e.g., putting a phone on a chair without it floating between the legs). Some participants preferred richer directional schemas (clock-face or cardinal points) over simple left/right cues, highlighting opportunities for more precise spatial references.

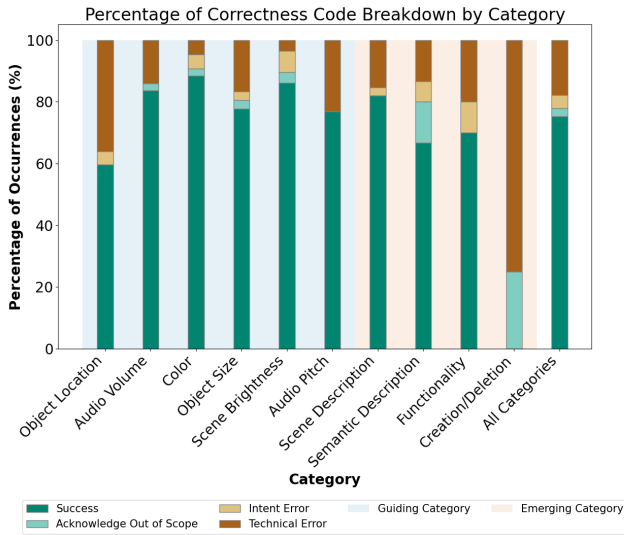
Audio Volume. Highly rated ($M = 4.8, SD = 0.5$), the Audio Volume category allowed participants to isolate and prioritize sounds. In Scene 2, for example, most participants adjusted the cats’ meows to identify differences. P3-VI described this as mirroring real-world strategies: “*try to isolate one thing at a time and listen.*” P4-LP suggested volume control could help prioritize critical in-game alerts.

Color. Frequent and successful ($M = 4.6, SD = 0.5$), Color category prompts supported object recognition and contrast adjustments. For low-vision participants, recoloring enhanced readability and visibility (e.g., P5-VI applied black-on-white contrast; P6-VI recolored objects to “*see them and move them and understand what’s there even better*”). For blind participants, relevance varied: P7-B noted limited utility without color knowledge, whereas P8-B, who had prior vision, used color references to mentally reconstruct scenes. Regarding potential improvements, some participants ($N=2$) suggested that the system should better recognize an object’s background color to select an appropriate high-contrast color.

Object Size. Moderately rated ($M = 4.1, SD = 0.8$), resizing aided low-vision players. P6-VI explained that enlarging objects made them easier to recognize and manipulate. For others, the numeric size descriptions drew mixed reactions. P7-B valued precise measurements in meters or feet, since these remain consistent regardless of viewing distance: “*as you move away from an object, it visually appears smaller. If everything is in meters or feet... then [the description] is not affected by scale.*” In contrast, P8-B found numeric values difficult to interpret without visual reference. Some participants proposed alternatives: P2-VI suggested zooming in on objects



(a) Occurrences of each category (blue = guiding, orange = emerging).



(b) Correctness breakdown per category.

Figure 5: Prompt usage and correctness across categories.

rather than resizing them, while P5-VI highlighted the importance of knowing size limits (e.g., maximum enlargement). Others, like P4-LP, considered size largely aesthetic and of limited accessibility value.

Scene Brightness. Less frequently used and variably rated ($M = 3.6, SD = 1.7$), brightness adjustments helped some low-vision participants by enhancing contrast (P6-VI) or highlighting local areas (P5-VI). For blind participants, value was limited. For example, P4-LP dismissed it as “cosmetic.” Usefulness thus depended strongly on visual ability.

Audio Pitch. Lowest rated ($M = 2.8, SD = 1.7$), Audio Pitch adjustments were less useful overall. Some participants envisioned benefits for distinguishing status (e.g., damage to a game character)

or orientation (P7-B: “using pitch for positional information is a really good idea... for example, gets higher when you get closer to it”), but most preferred volume as a more salient cue. Participant comments suggested potential for system-level pitch cues for important events or orientation in future designs, but this category had limited value as a user-driven adjustment in our scenarios.

5.3.2 Emerging Categories.

Scene and Semantic Descriptions. Although not predefined, these categories were frequently used, revealing additional details preferred by participants. Scene descriptions offered general overviews, while semantic descriptions probed higher-level qualities (e.g., the mood of a cat or the content of a laptop screen). Participants valued detail but noted issues of verbosity: P4-LP suggested limiting output to nearby objects or allowing control over granularity. P7-B emphasized context-aware tailoring: “Maybe all I need to know is... I don’t need to know that it’s a Toshiba.” Out-of-scope acknowledgments were common, as users often sought knowledge beyond system capabilities. Suggested improvements included adding shape (P1-LP), clearance (P7-B), sound characteristics (P2-VI, P8-B), and hazard indicators (P7-B).

Functionality and Creation/Deletion. Although infrequent, these categories revealed unmet needs. Participants wanted to interact with objects functionally (e.g., “sit on the bench” rather than just move to its coordinates) and to add or remove objects. As P4-LP pointed out, without supporting object functionality interaction, the objects felt like “dry artifacts” instead of interactive video game objects. Success rates were low, especially for Creation/Deletion, reflecting system limitations rather than lack of interest. However, these attempts highlight aspiration for richer interactivity beyond current system capabilities.

5.3.3 Category Usage Comparison Between Scene 2 and Scene 3. Scene 2 asked participants to complete specific tasks aligned with the six guiding categories, whereas Scene 3 encouraged open-ended exploration. Figure 6 shows the percentage of prompts in each category for the two scenes (excluding *Compound* and *Other* prompts). Five categories showed notably higher relative usage in Scene 3 than in Scene 2: Object Location, Scene Description, Audio Pitch, Functionality, and Creation/Deletion. These differences suggest two broad exploration patterns in the open-ended scene: *spatial exploration* (Object Location, Scene Description) and *novel feature exploration* (Audio Pitch, Functionality, Creation/Deletion). Together, they highlight how participants appropriated RAVEN differently when not guided by predefined tasks.

5.4 Emergent User Strategies and Goals

Across both the task-oriented scene (Scene 2) and the open-ended scene (Scene 3), participants used prompts serving different goals. Our coding produced eight user goal codes, grouped into three categories: *exploration*, *execution*, and *verification* (see table 5 for details and examples).

Exploration. Exploration was the entry point for nearly all tasks. It included US (understand scene), SS (search in scene), and QI (questions about specific items). In Scene 2, 33 of 48 tasks (68.8%) began with exploration, and in Scene 3, all eight participants opened

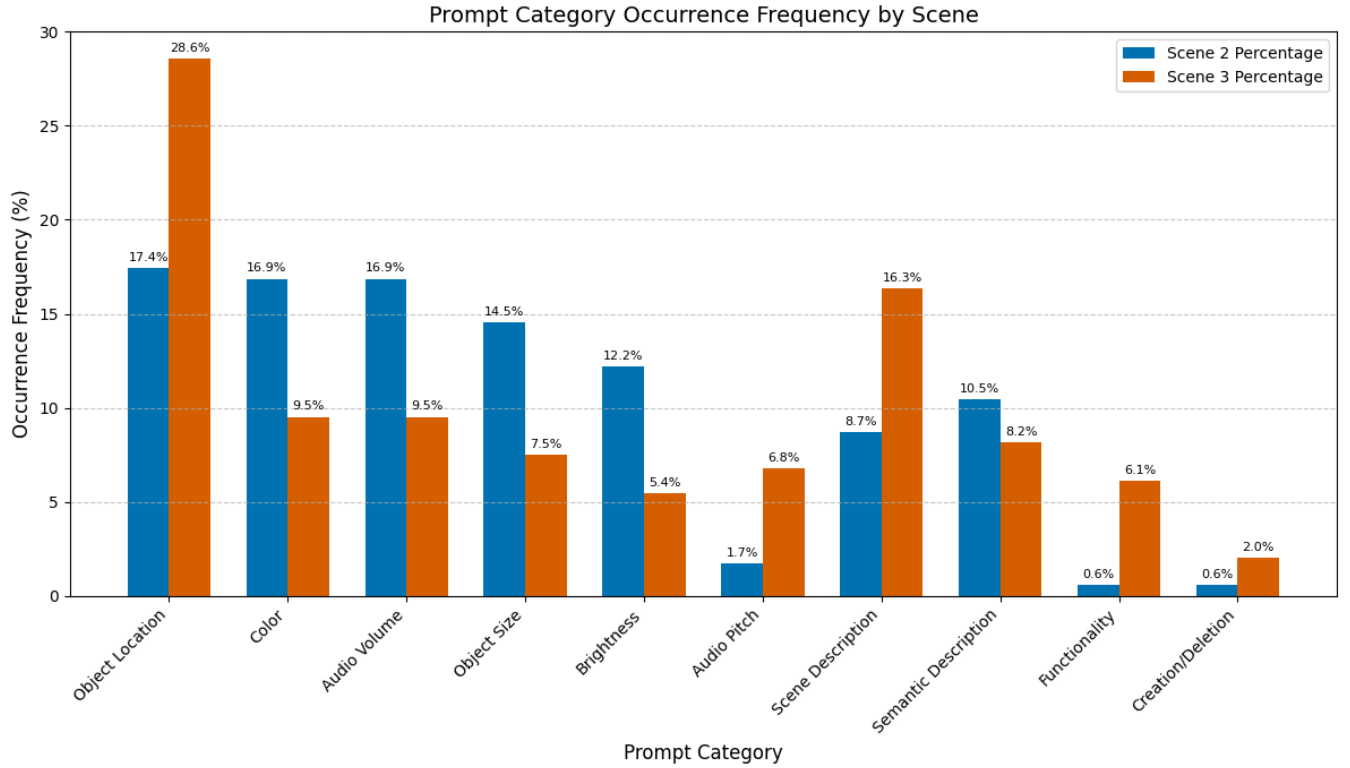


Figure 6: Occurrence frequency of each category within the categorizable prompts (excluding *Compound* and *Other*) in each scene (blue = Scene 2 percentage, orange = Scene 3 percentage).

with such prompts. P5-VI explained: “*You have to first ask what is an object, then know what it is, then increase size... and be more specific.*” Among the 336 total prompts, QI was most frequent (93), followed by SS (48) and US (36). These patterns highlight the central role of exploration in helping BLV users build an initial mental model of the environment. The success rate for US prompts (86.1%) was higher than for SS (75.0%) and QI (73.1%), suggesting that strategies starting with broad, whole-scene queries were more reliably supported than strategies that immediately targeted specific objects.

Execution. After establishing orientation, participants used execution prompts to act on the scene. Codes included EK (external knowledge), EM (explicit modification), PM (proactive modification), and CM (creative modification). EK was rare (7 prompts) but notable, as it showed users using the LLM to probe for accessibility knowledge beyond the scene itself, such as “*If I had a camera, would the light sources be bright enough to take a clear picture?*” P4-LP highlighted its promise: “*you could have the game intuit its own design and say how might this be more accessible in a non-visual way?*” EK prompts also demonstrated a relatively high success rate (85.7%), underscoring that this strategy worked well when participants treated the LLM as an accessibility advisor.

EM and PM were common, aligning with task completion and proactive adjustments. Interestingly, CM appeared in 34 prompts, with participants using the system creatively, such as “*Create a door in front of me*” (P2-VI) or “*add a canopy and luxury padding to the*

bench” (P7-B). For some, creative use became dominant — P2-VI devoted 10 of her 17 free-exploration prompts to CM. Creative modifications were sometimes successful but also more likely to expose limitations (e.g., unsupported functionality or implausible object behavior), indicating that CM is a powerful but less predictable strategy compared to EM and PM. These findings suggest that beyond accessibility, participants viewed the system as a potential authoring tool.

Verification. Verification prompts (V) were used to confirm whether modifications occurred as intended. They appeared 37 times overall, primarily in Scene 2 where tasks had specific targets. In Scene 3, use varied: some participants (N=5) regularly checked outcomes, while others (N=3) did not verify at all. P8-B issued six verification prompts across twelve modifications, explaining: “*software applications or technology is not perfect, and that you just [have to] understand that there will be glitches.*” This reflects differing trust strategies: some relied on system feedback directly, while others double-checked modifications through dialogue.

6 Preliminary Evaluation with Unity Developers

6.1 Method

To understand how RAVEN fits into real development workflows and to evaluate its scalability from a developer perspective, we

Table 5: User goal codes across three groups.

Group	Code	Description	Example
Exploration	US	Understand whole scene	“What is this scene like?”
Exploration	SS	Search within scene	“Is there a white cat?”, “Where is the loudest cat?”
Exploration	QI	Question about specific item(s)	“How big is the conference table?”
Execution	EK	External knowledge	“What color is useful for low-vision people?”
Execution	EM	Explicit modification (task-driven)	“Make the bench bigger.”
Execution	PM	Proactive modification (user-driven)	“Mute the other cats.”
Execution	CM	Creative modification	“Make the chairs float in the air.”
Verification	V	Verify changes	“What’s the color of the chair now?”

conducted a preliminary usability study with six Unity developers (three men, three women). Participants were recruited through forum posts and snowball sampling. Their Unity experience ranged from 0.5 to 6 years ($M=1.8$, $SD=1.9$). Two participants had prior experience with accessibility toolkits (e.g., Meta XR accessibility tools), and two had used LLMs in game development.

Researchers first demonstrated how to integrate RAVEN into a simple example Unity scene (five objects including two 3D objects, a sound source, a light source, and a text object) following the workflow described in section 3.3.1. Participants then repeated this integration themselves in the example scene. Next, they incorporated RAVEN into one of their own scenes, either self-created or selected from online resources. Written instructions accompanied the implementation steps, and participants were not given time constraints.

We recorded (1) the time required to achieve a working integration in the example scene and (2) the time needed to apply a functioning version of RAVEN to a personal scene. Time was measured from the start of implementation to the first successful runtime test of an LLM prompt. Although we did not explicitly prompt developers to think aloud, several spontaneously verbalized thoughts about the system during the process. Because some personal scenes were large, we also collected estimates of how many objects would require tagging for full coverage.

After implementation, participants completed a questionnaire (5-point Likert-scale items on learnability, usability, perceived accessibility improvement, and intent for future use), followed by a semi-structured interview to capture experiences, preferences, and improvement suggestions. Sessions were conducted in person, lasted approximately one hour, and participants received \$30 compensation. Quantitative ratings were summarized using descriptive statistics. Two coders performed thematic analysis [34], producing 12 codes grouped under four themes—initial learning, developer interaction, system behavior, and perceived accessibility impact (see supplementary materials for full code book).

6.2 Findings

Developers spent an average of 265.8 seconds ($SD=92.6$) integrating RAVEN into the example scene and 331.8 seconds ($SD=63.2$) applying a working version to their own scenes. Estimated tagging effort averaged 26.7 objects per scene ($SD=16.7$), though estimates varied with scene complexity and the tagging strategies developers preferred.

Initial Learning and Setup. Developers described the setup process as easy and intuitive. Learnability received a mean rating of 4.7 ($SD=0.5$). All six participants commented on the ease of initial system setup. Three appreciated the drag-and-drop design, while two noted that such simplicity could particularly help novices who may feel overwhelmed by other accessibility tools. A recurring challenge involved unclear variable names (e.g., using `isMeta` to denote non-physical objects like sunlight); four participants recommended more transparent naming conventions.

Developer Interaction and Tagging Strategies. Usability ratings remained positive when applying RAVEN to developers’ own scenes ($M=4.3$, $SD=0.5$). Tagging strategies varied: some developers preferred tagging only interactable objects, while others advocated tagging everything visible for parity with sighted access. D4 noted that tagging hidden or narrative-sensitive elements (e.g., easter eggs) might unintentionally spoil gameplay, whereas D5 suggested including a tagging field to mark the importance of each object. D6 argued that “a sighted person can see all the objects, so to provide an equivalent experience, we should tag all of them.”

System Behavior and Desired Automation. Developers suggested automating several aspects of the workflow, including: generating unique object names (D2), enabling batch tagging (D3), collecting renderers from nested objects (D4), and generating text descriptions from models or images (D6). D2 also suggested an interface for customizing model and prompt settings, though this might increase the learning curve.

Perceived Accessibility Impact and Adoption. Developers rated RAVEN’s potential to improve accessibility highly ($M=4.7$, $SD=0.5$) and said they would use it in future projects ($M=4.7$, $S=0.5$). Five participants felt RAVEN could meaningfully enhance BLV user experience. Interestingly, D3 described the system as a connector between developers and disabled users: developers supply descriptions and structural intent, while the LLM tailors them to user needs.

However, three participants raised concerns about LLM-generated errors, echoing concerns from prior work in runtime generation [39] and feedback from BLV participants. For future improvements, participants suggested features such as event-history tracking (D6), improved input/output methods (N=4), and using RAVEN not only for modifying scenes but also for authoring accessible scenes (N=2).

7 Discussion

Generative AI introduces new programming paradigms and conversational forms of human-computer interaction. RAVEN illustrates how these capabilities can extend accessibility in 3D environments by allowing BLV users to query and adapt scenes in natural language. Our findings show that participants found the system usable and intuitive, valuing its flexibility and tolerance of ambiguous input. At the same time, accuracy limitations surfaced, raising questions of trust and verification—findings consistent with known limitations of LLMs for accessibility [1, 28].

Throughout the study, participants appropriated RAVEN in different ways across Scenes 2 and 3, highlighting both task-driven use and open-ended exploration. These behaviors, together with the error profile we observed, inform how RAVEN should be positioned relative to existing accessibility tools and how future systems might provide safe, scalable support. Building on these results, we discuss opportunities for safeguarding and verification, the potential role of LLMs as accessibility experts, ways to reduce developer effort through metadata automation, and broader implications for conversational programming. We conclude with limitations and directions for future work.

7.1 Towards Safe and Trustworthy On-Request Generative Access

Open-ended prompting enabled BLV participants to make diverse runtime modifications, but its open-endedness also created risks. Prior work in runtime behavior generation has noted developer concerns about “game-breaking” mechanics, such as deleting important objects or blocking pathways [39]. Our findings echo these risks: four emergent categories of prompts went beyond expected usage, often resulting in hallucinations or failures. Compounded by the difficulty BLV participants faced in verifying modifications, these results underscore the need for both automated and user-driven safeguards.

Accessibility-Focused Guardrails. Prior work has proposed multi-layer guardrails to constrain LLM behavior across domains [20]. Extending this approach, accessibility-focused guardrails could ensure that generated modifications preserve both functionality and accessibility. Future work might adapt existing secure code-generation frameworks such as static analysis and constraint-based filters (e.g., CodeShield [20]) to restrict modifications that could regress accessibility, building on recent guidelines for accessible agentic interaction [24, 61, 92].

Automated Verification. In our study, verification relied on the same LLM that generated modifications, risking false confirmations when errors occurred. Future systems could mitigate this by separating generation and verification roles, using multi-agent methods such as ensemble consensus [55, 86] or debate-based techniques [73]. Multi-modal models could further enhance verification by cross-checking code execution against visual and auditory outputs from the scene [31]. Though GPT-4o supports such multi-modal capabilities, we were unable to use them as these features were not available in the APIs at system development time.

Human-in-the-Loop Verification. Participants frequently issued follow-up prompts to confirm whether modifications had been applied, reflecting both natural interaction patterns and limited

trust in the system. Prior work shows that LLMs can only audit accessibility in a limited capacity in well-standardized domains such as apps [96] and the web [60], underscoring the need for human oversight in open-ended contexts like 3D environments. Building on collaborative accessibility approaches in mainstream platforms (e.g., Xbox’s controller assist), future systems could weave accessibility verification into mixed-ability collaborative gameplay. Such designs would preserve BLV user agency [5, 56] while distributing responsibility for accuracy between users and systems, making accessibility verification an interdependent experience in virtual worlds [9, 13].

Taken together, guardrails, automated checks, and human oversight point to a layered verification strategy for ensuring both reliability and user trust in generative accessibility systems.

7.2 LLMs as Accessibility Experts

Participants sometimes prompted the system as if it were an accessibility consultant, asking it to recommend or directly apply accessibility improvements (e.g., “make the bench more visible to visually impaired individuals”). This highlights users’ expectation that LLMs can provide design knowledge beyond simple scene modifications. Prior work shows that LLMs can support accessibility tasks in domains such as web and app design [1, 28], but also that they risk introducing ableist assumptions or biased recommendations. In 3D environments, this challenge is compounded by the need to scope recommendations to both user ability and scene context. Future systems could integrate guardrails that align suggestions with achievable in-system capabilities and contextual constraints, enabling LLMs to act as reliable accessibility advisors rather than overgeneralizing.

7.3 Positioning RAVEN Among Accessibility Approaches

RAVEN occupies a distinct space among existing accessibility approaches for virtual 3D environments. Static toolkits such as SeeingVR [95] offer valuable enhancements for low-vision users but rely on developer-authored overlays and visual filters. These toolkits do not support runtime code generation, natural language prompting, or modification of audio and spatial structures. As a result, they are complementary to rather than directly comparable with RAVEN, which focuses on enabling BLV users to interactively query and modify 3D environments in real time.

An alternative design question is whether large language models could simply “make the entire scene accessible” in a single transformation. Our findings suggest that such one-shot global modifications would not meet BLV users’ needs. Participants expressed diverse and sometimes conflicting accessibility preferences across the visual-ability spectrum—for example, some preferred brighter scenes while others needed dimmer ones, and some relied on color semantics while others valued purely spatial descriptions. Participants also engaged in iterative exploration, targeted adjustments, and verification, using conversational interaction to personalize accessibility in ways that a single global transformation could not capture. Importantly, accessibility needs emerged as highly individualized and context-dependent in our study, suggesting that

adaptive, user-guided modification workflows are more appropriate than rigid, global transformations.

By enabling controlled, query-guided modifications grounded in embodied metadata and accessibility rules, RAVEN supports accessible interaction as an iterative, user-directed process. This design preserves user agency, aligns modifications with contextual grounding, and avoids the brittleness of one-shot transformations, positioning RAVEN as a complementary and necessary approach alongside existing static and LLM-based accessibility tools.

7.4 Reducing Developer Burden through Metadata Automation

Our system required manual developer setup to populate the semantic scene graph with object and sound descriptions (section 6). In our preliminary developer study, participants found the overall workflow learnable and usable, but expressed concern about scaling tagging and description efforts to larger scenes. Several requested automation to generate names, collect renderers from nested objects, and produce first-pass descriptions for both visual and audio elements.

These requested capabilities can be supported in future integrations of RAVEN’s authoring into mainstream developer environments. For example, Cap3D provides scalable 3D object captioning [48], and ExCap3D supports expressive, variable-level descriptions [88]. Audio language models (e.g., Audio Flamingo [43]) can generate captions for sound sources. Accessibility-focused plugins such as UI Accessibility Plugin [52] could integrate these intelligent capabilities to generate a first-pass layer of accessibility metadata for developers.

Future systems could integrate such tools into development workflows (e.g., Unity editor scripting [78]) to semi-automate tagging. Developers would still refine outputs to capture context-specific meaning (e.g., labeling a “red vial” as a “health potion”), but automation could substantially reduce workload. By combining automated metadata generation with human curation, systems could better support detailed, context-aware accessibility queries without placing unrealistic demands on developers.

7.5 Conversational Programming for Personalized Accessibility

Beyond ensuring safety and scalability, our findings highlight the importance of personalization—how accessibility preferences differ across users and can be supported through conversational programming. RAVEN is, to our knowledge, the first system to combine conversational interaction with runtime code generation for accessibility in 3D environments. Despite current limitations, participants’ engagement demonstrates the potential of conversational programming [65] to remediate accessibility barriers in ways that adapt to individual preferences, particularly where fixed guidelines fall short.

Our findings showed marked differences in how categories were valued depending on participants’ visual ability and history of vision. Low-vision participants benefited from color and brightness adjustments, while blind participants often dismissed these as cosmetic or irrelevant. Conversely, blind participants emphasized precise spatial orientation, whereas some low-vision participants

preferred richer visual cues. By accommodating both, RAVEN responds to the diverse and sometimes conflicting preferences of BLV users [75], demonstrating how conversational programming can enable personalized accessibility. Rather than treating “blind” and “low-vision” users as two homogeneous groups, our analysis reflects this continuum of abilities and preferences, which further motivates an adaptive, user-directed approach over static presets.

Beyond 3D environments, similar approaches could extend to domains such as web browsing, education, or productivity tools. For example, extensible screen readers like NVDA could be augmented with LLM capabilities to generate custom add-ons from natural language instructions, tailoring themselves to individual needs. More broadly, the non-prescriptive and adaptive nature of conversational programming could support people with multiple disabilities or with fluctuating access needs, where requirements change across contexts and personal circumstances [49].

7.6 Limitations and Future Work

Our study represents an early exploration of conversational programming for accessibility in 3D environments. Several limitations qualify our findings and point to opportunities for future work. First, the system exhibited a non-trivial error rate, meaning it is not yet deployable in real-world applications and may have negatively impacted user trust and usability during evaluation. Future work should prioritize error prevention and robust verification methods to increase trust and reliability. Second, the system lacked an understanding of object affordances (e.g., distinguishing the sitting surface of a bench from its coordinates), limiting functional interactions. Richer semantic modeling of object properties will be needed to support more realistic accessibility modifications. Third, our evaluation was conducted in a controlled, scenario-based setting with short-term tasks. Although this is comparable to other accessibility evaluations, our sample size (eight BLV participants) limits the statistical power of between-group comparisons, which means our findings should be interpreted primarily as qualitative and exploratory. Finally, longer-term deployments in commercial games or fast-paced contexts (e.g., combat scenarios) are necessary to assess real-world viability.

8 Conclusion

RAVEN explores a new frontier in accessible interaction—enabling blind and low-vision users to query and modify 3D virtual environments through natural language. By combining semantic scene understanding with real-time code generation, RAVEN shifts accessibility from a static, developer-defined feature to an interactive, user-driven experience. Our evaluation with eight BLV participants demonstrated the promise of this approach: users found the system intuitive, flexible, and empowering. At the same time, limitations around reliability, error transparency, and emergent user needs point to the importance of future advances in guardrails, verification, and automated metadata. Such efforts will be critical to ensuring safe, trustworthy, and scalable deployment of generative accessibility systems.

References

- [1] Rudaiba Adnin and Maitraye Das. 2024. “I look at it as the king of knowledge”: How Blind People Use and Understand Generative AI Tools. In *Proceedings of*

- the 26th International ACM SIGACCESS Conference on Computers and Accessibility (New York, NY, USA, 2024-10-27) (ASSETS '24). Association for Computing Machinery, 1–14. doi:10.1145/3663548.3675631
- [2] Rahaf Alharbi, Pa Lor, Jaylin Herskovitz, Sarita Schoenebeck, and Robin N. Brewer. 2024. Misfitting With AI: How Blind People Verify and Contest AI Errors. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) (ASSETS '24). Association for Computing Machinery, New York, NY, USA, Article 61, 17 pages. doi:10.1145/3663548.3675659
 - [3] Wajdi Aljedaani, Abdulrahman Habib, Ahmed Aljohani, Marcelo Eler, and Yunhe Feng. 2024. Does ChatGPT Generate Accessible Code? Investigating Accessibility Challenges in LLM-Generated Source Code. In *Proceedings of the 21st International Web for All Conference* (Singapore, Singapore) (W4A '24). Association for Computing Machinery, New York, NY, USA, 165–176. doi:10.1145/3677846.3677854
 - [4] Ronny Andrade, Steven Baker, Jenny Waycott, and Frank Vetere. 2018. Echo-house: exploring a virtual environment by using echolocation. In *Proceedings of the 30th Australian Conference on Computer-Human Interaction* (Melbourne, Australia) (OzCHI '18). Association for Computing Machinery, New York, NY, USA, 278–289. doi:10.1145/3292147.3292163
 - [5] Harshadha Balasubramanian, Cecily Morrison, Martin Grayson, Zhanat Makhataeva, Rita Faia Marques, Thomas Gable, Dalya Perez, and Edward Cutrell. 2023. Enable Blind Users' Experience in 3D Virtual Environments: The Scene Weaver Prototype. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2023-04-19) (CHI EA '23). Association for Computing Machinery, 1–4. doi:10.1145/3544549.3583909
 - [6] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: adding an adjective rating scale. *J. Usability Studies* 4, 3 (May 2009), 114–123.
 - [7] Ben Bayliss. 2022. *Fortnite Accessibility — Menu Deep Dive*. <https://caniplaythat.com/2022/04/26/fortnite-accessibility-menu-deep-dive/>
 - [8] Be My Eyes 2025. *Introducing: Be My AI*. Retrieved Jan 24, 2025 from <https://www.bemyeyes.com/blog/introducing-be-my-ai>
 - [9] Cynthia L. Bennett, Erin Brady, and Stacy M. Branham. 2018. Interdependence as a Frame for Assistive Technology Research and Design. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility* (Galway, Ireland) (ASSETS '18). Association for Computing Machinery, New York, NY, USA, 161–173. doi:10.1145/3234695.3236348
 - [10] Cynthia L. Bennett, Renee Shelby, Negar Rostamzadeh, and Shaun K. Kane. 2024. Painting with Cameras and Drawing with Text: AI Use in Accessible Creativity. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (New York, NY, USA, 2024-10-27) (ASSETS '24). Association for Computing Machinery, 1–19. doi:10.1145/3663548.3675644
 - [11] Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C. Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, Mark Ibrahim, Melissa Hall, Yuniang Xiong, Jonathan Lebensold, Candace Ross, Srihari Jayakumar, Chuan Guo, Diane Bouchacourt, Haider Al-Tahan, Karthik Padthe, Vasu Sharma, Hu Xu, Xiaoqing Ellen Tan, Megan Richards, Samuel Lavoie, Pietro Astolfi, Reyhane Askari Hemmat, Jun Chen, Kushal Tirumala, Rim Assouel, Mazda Moayeri, Arjang Talattof, Kamalika Chaudhuri, Zechun Liu, Xilun Chen, Quentin Garrido, Karen Ullrich, Aishwarya Agrawal, Kate Saenko, Asli Celikyilmaz, and Vikas Chandra. 2024. An Introduction to Vision-Language Modeling. arXiv:2405.17247 [cs.LG] <https://arxiv.org/abs/2405.17247>
 - [12] Doug A. Bowman, Chris North, Jian Chen, Nicholas F. Polys, Pardha S. Pyla, and Umur Yilmaz. 2003. Information-rich virtual environments: theory, tools, and research agenda. In *Proceedings of the ACM symposium on Virtual reality software and technology* (New York, NY, USA, 2003-10-01) (VRST '03). Association for Computing Machinery, 81–90. doi:10.1145/1008653.1008669
 - [13] Stacy M. Branham and Shaun K. Kane. 2015. Collaborative Accessibility: How Blind and Sighted Companions Co-Create Accessible Home Spaces. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) (CHI '15). Association for Computing Machinery, New York, NY, USA, 2373–2382. doi:10.1145/2702123.2702511
 - [14] John Brooke. 1996. SUS: A 'Quick and Dirty' Usability Scale. In *Usability Evaluation In Industry*. CRC Press. Num Pages: 6.
 - [15] Xinyun Cao, Kexin Phyllis Ju, Chenglin Li, Venkatesh Potluri, and Dhruv Jain. 2025. Demo of RAVEN: Realtime Accessibility in Virtual Environments for Blind and Low-Vision People. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility* (ASSETS '25). Association for Computing Machinery, New York, NY, USA, Article 168, 5 pages. doi:10.1145/3663547.3759725
 - [16] Ruei-Che Chang, Chia-Sheng Hung, Bing-Yu Chen, Dhruv Jain, and Anhong Guo. 2024. SoundShift: Exploring Sound Manipulations for Accessible Mixed-Reality Awareness. In *Proceedings of the 2024 ACM Designing Interactive Systems Conference* (Copenhagen, Denmark) (DIS '24). Association for Computing Machinery, New York, NY, USA, 116–132. doi:10.1145/3643834.3661556
 - [17] Ruei-Che Chang, Yuxuan Liu, and Anhong Guo. 2024. WorldScribe: Towards Context-Aware Live Visual Descriptions. 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 140, 18 pages. doi:10.1145/3654777.3676375
 - [18] Ruei-Che Chang, Yuxuan Liu, Lotus Zhang, and Anhong Guo. 2024. EditScribe: Non-Visual Image Editing with Natural Language Verification Loops. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) (ASSETS '24). Association for Computing Machinery, New York, NY, USA, Article 65, 19 pages. doi:10.1145/3663548.3675599
 - [19] Junlong Chen, Jens Grubert, and Per Ola Kristensson. 2025. Analyzing Multimodal Interaction Strategies for LLM-Assisted Manipulation of 3D Scenes. In *2025 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. 206–216. doi:10.1109/VR59515.2025.00045
 - [20] Sahana Chennabasappa, Cyrus Nikolaidis, Daniel Song, David Molnar, Stephanie Ding, Shengye Wan, Spencer Whitman, Lauren Deason, Nicholas Doucette, Abraham Montilla, Alekhya Gampa, Beto de Paola, Dominik Gabi, James Crnkovich, Jean-Christophe Testud, Kat He, Rashnil Chaturvedi, Wu Zhou, and Joshua Saxe. 2025. LlamaFirewall: An open source guardrail system for building secure AI agents. arXiv:2505.03574 [cs.CR] <https://arxiv.org/abs/2505.03574>
 - [21] Chris Creed, Maadh Al-Kalbani, Arthur Theil, Sayan Sarcar, and Ian Williams. 2024. Inclusive AR/VR: Accessibility Barriers for Immersive Technologies. 23, 1 (2024), 59–73. doi:10.1007/s10209-023-00969-0 arXiv:2304.13465 [cs]
 - [22] Fernanda De La Torre, Cathy Mengying Fang, Han Huang, Andrzej Banburski-Fahey, Judith Amores Fernandez, and Jaron Lanier. 2024. LLMR: Real-time Prompting of Interactive Worlds using Large Language Models. 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 600, 22 pages. doi:10.1145/3613904.3642579
 - [23] Jon E. Froehlich, Alexander J. Fiannaca, Nimer M. Jaber, Victor T saran, and Shaun K. Kane. 2025. StreetViewAI: Making Street View Accessible Using Context-Aware Multimodal AI. In *Proceedings of the 38th Annual ACM Symposium on User Interface Software and Technology* (UIST '25). Association for Computing Machinery, New York, NY, USA, Article 43, 22 pages. doi:10.1145/3746059.3747756
 - [24] Biying Fu, Abdenour Hadid, and Naser Damer. 2025. Generative AI in the context of assistive technologies: Trends, limitations and future directions. 154 (2025), 105347. doi:10.1016/j.imavis.2024.105347
 - [25] Diogo Furtado, Renato Alexandre Ribeiro, Manuel Piçarra, Leticia Seixas Pereira, Carlos Duarte, André Rodrigues, and João Guerreiro. 2025. Designing and Evaluating a VR Boxing Experience with Blind People. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems* (CHI '25). Association for Computing Machinery, New York, NY, USA, Article 1246, 17 pages. doi:10.1145/3706598.3713374
 - [26] Vinitha Gadiraju, Shaun Kane, Sunipa Dev, Alex Taylor, Ding Wang, Remi Denton, and Robin Brewer. 2023. "I wouldn't say offensive but...": Disability-Centered Perspectives on Large Language Models. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency* (Chicago, IL, USA) (FAccT '23). Association for Computing Machinery, New York, NY, USA, 205–216. doi:10.1145/3593013.3593989
 - [27] Kate Glazko, Yusuf Mohammed, Ben Kosa, Venkatesh Potluri, and Jennifer Mankoff. 2024. Identifying and Improving Disability Bias in GPT-Based Resume Screening. In *Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency* (Rio de Janeiro, Brazil) (FAccT '24). Association for Computing Machinery, New York, NY, USA, 687–700. doi:10.1145/3630106.3658933
 - [28] Kate S Glazko, Momona Yamagami, Aashaka Desai, Kelly Avery Mack, Venkatesh Potluri, Xuhai Xu, and Jennifer Mankoff. 2023. An Autoethnographic Case Study of Generative Artificial Intelligence's Utility for Accessibility. In *Proceedings of the 25th International ACM SIGACCESS Conference on Computers and Accessibility* (New York, NY, USA) (ASSETS '23). Association for Computing Machinery, New York, NY, USA, Article 99, 8 pages. doi:10.1145/3597638.3614548
 - [29] Aaron Gluck, Kwajo Boateng, and Julian Brinkley. 2021. Racing in the Dark: Exploring Accessible Virtual Reality by Developing a Racing Game for People who are Blind. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting* 65, 1 (2021), 1114–1118. doi:10.1177/1071181321651224 arXiv:https://doi.org/10.1177/1071181321651224
 - [30] J.L. Gonzalez-Mora, A. Rodriguez-Hernandez, E. Burunat, F. Martin, and M.A. Castellano. 2006. Seeing the world by hearing: Virtual Acoustic Space (VAS) a new space perception system for blind people.. In *2006 2nd International Conference on Information & Communication Technologies* (2006-04), Vol. 1. 837–842. doi:10.1109/ICTTA.2006.1684482
 - [31] Google. 2025. *Live API | Gemini API*. Retrieved June 08, 2025 from <https://ai.google.dev/gemini-api/docs/live>
 - [32] William Grussenmeyer and Elke Folmer. 2017. Accessible Touchscreen Technology for People with Visual Impairments: A Survey. *ACM Trans. Access. Comput.* 9, 2, Article 6 (Jan. 2017), 31 pages. doi:10.1145/3022701
 - [33] João Guerreiro, Yujin Kim, Rodrigo Nogueira, SeungA Chung, André Rodrigues, and Uran Oh. 2023. The Design Space of the Auditory Representation of Objects and Their Behaviours in Virtual Reality for Blind People. *IEEE Transactions on Visualization and Computer Graphics* 29, 5 (2023), 2763–2773. doi:10.1109/TVCG.2023.3247094
 - [34] Greg Guest, Kathleen M. MacQueen, and Emily E. Namey. 2012. *Applied Thematic Analysis*. SAGE Publications, Inc. doi:10.4135/9781483384436

- [35] Xiyun Hu, Dizhi Ma, Fengming He, Zhengzhe Zhu, Shao-Kang Hsia, Chenfei Zhu, Ziyi Liu, and Karthik Ramani. 2025. GesPrompt: Leveraging Co-Speech Gestures to Augment LLM-Based Interaction in Virtual Reality. In *Proceedings of the 2025 ACM Designing Interactive Systems Conference (DIS '25)*. Association for Computing Machinery, New York, NY, USA, 59–80. doi:10.1145/3715336.3735769
- [36] Ziniu Hu, Ahmet Iscen, Aashi Jain, Thomas Kipf, Yisong Yue, David A Ross, Cordelia Schmid, and Alireza Fathi. 2024. SceneCraft: An LLM Agent for Synthesizing 3D Scenes as Blender Code. In *Forty-first International Conference on Machine Learning*. <https://openreview.net/forum?id=gAyzjHw2ml>
- [37] Mina Huh, Yi-Hao Peng, and Amy Pavel. 2023. GenAssist: Making Image Generation Accessible. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology* (San Francisco, CA, USA) (UIST '23). Association for Computing Machinery, New York, NY, USA, Article 38, 17 pages. doi:10.1145/3586183.3606735
- [38] G. Jansson, H. Petrie, C. Colwell, D. Kornbrot, J. Fänger, H. König, K. Billberger, A. Hardwick, and S. Furner. 1999. Haptic Virtual Environments for Blind People: Exploratory Experiments with Two Devices. 4, 1 (1999), 8–17. doi:10.20870/IJVR.1999.4.1.2663 Number: 1.
- [39] Nicholas Jennings, Han Wang, Isabel Li, James Smith, and Bjoern Hartmann. 2024. What's the Game, then? Opportunities and Challenges for Runtime Behavior Generation. In *Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology* (New York, NY, USA, 2024-10-11) (UIST '24). Association for Computing Machinery, 1–13. doi:10.1145/3654777.3676358
- [40] Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Lukas Vierling, Donghai Hong, Jiayi Zhou, Zhaowei Zhang, Fanzhi Zeng, Juntao Dai, Xuehai Pan, Kwan Yee Ng, Aidan O'Gara, Hua Xu, Brian Tse, Jie Fu, Stephen McAleer, Yaodong Yang, Yizhou Wang, Song-Chun Zhu, Yike Guo, and Wen Gao. 2025. AI Alignment: A Comprehensive Survey. arXiv:2310.19852 [cs.AI] <https://arxiv.org/abs/2310.19852>
- [41] Sanchita S. Kamath, Aziz Zeidieh, Omar Khan, Dhruv Sethi, and JooYoung Seo. 2024. Playing Without Barriers: Crafting Playful and Accessible VR Table-Tennis with and for Blind and Low-Vision Individuals. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) (ASSETS '24). Association for Computing Machinery, New York, NY, USA, Article 88, 5 pages. doi:10.1145/3663548.3688526
- [42] Satwik Ram Kodandaram, Uttu Uckun, Xiaojun Bi, IV Ramakrishnan, and Vikas Ashok. 2024. Enabling Uniform Computer Interaction Experience for Blind Users through Large Language Models. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) (ASSETS '24). Association for Computing Machinery, New York, NY, USA, Article 73, 14 pages. doi:10.1145/3663548.3675605
- [43] Zhifeng Kong, Arushi Goel, Rohan Badlani, Wei Ping, Rafael Valle, and Bryan Catanzaro. 2024. Audio Flamingo: A Novel Audio Language Model with Few-Shot Learning and Dialogue Abilities. arXiv:2402.01831 [cs, eess] <http://arxiv.org/abs/2402.01831>
- [44] Julian Kreimeier and Timo Götzmann. 2020. Two Decades of Touchable and Walkable Virtual Reality for Blind and Visually Impaired People: A High-Level Taxonomy. *Multimodal Technologies and Interaction* 4, 4 (2020). doi:10.3390/mti4040079
- [45] Seonghee Lee, Maho Kohga, Steve Landau, Sile O'Modhrain, and Hari Subramanyam. 2024. AltCanvas: A Tile-Based Editor for Visual Content Creation with Generative AI for Blind or Visually Impaired People. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) (ASSETS '24). Association for Computing Machinery, New York, NY, USA, Article 70, 22 pages. doi:10.1145/3663548.3675600
- [46] Chen Liang, Yuxuan Liu, Martez Mott, and Anhong Guo. 2025. HandProxy: Expanding the Affordances of Speech Interfaces in Immersive Environments with a Virtual Proxy Hand. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 9, 3, Article 107 (Sept. 2025), 30 pages. doi:10.1145/3749484
- [47] Cher P. Lim, Darren Nonis, and John Hedberg. 2006. Gaming in a 3D multiuser virtual environment: engaging students in Science lessons. 37, 2 (2006), 211–231. doi:10.1111/j.1467-8535.2006.00531.x eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8535.2006.00531.x>
- [48] Tiange Luo, Chris Rockwell, Honglak Lee, and Justin Johnson. 2023. Scalable 3D Captioning with Pretrained Models. In *Advances in Neural Information Processing Systems* (2023), A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (Eds.), Vol. 36. Curran Associates, Inc., 75307–75337. https://proceedings.neurips.cc/paper_files/paper/2023/file/ee4814f9bce0cae7991d3341bb081b55-Paper-Datasets_and_Benchmarks.pdf
- [49] Kelly Mack, Emma J. McDonnell, Leah Findlater, and Heather D. Evans. 2022. Chronically Under-Addressed: Considerations for HCI Accessibility Practice with Chronically Ill People. In *Proceedings of the 24th International ACM SIGACCESS Conference on Computers and Accessibility* (Athens, Greece) (ASSETS '22). Association for Computing Machinery, New York, NY, USA, Article 9, 15 pages. doi:10.1145/3517428.3544803
- [50] Kelly Avery Mack, Rida Qadri, Remi Denton, Shaun K. Kane, and Cynthia L. Bennett. 2024. “They only care to show us the wheelchair”: disability representation in text-to-image AI models. 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 288, 23 pages. doi:10.1145/3613904.3642166
- [51] Shachar Maidenbaum, Galit Buchs, Sami Abboud, Ori Lavi-Rotbain, and Amir Amedi. 2016. Perception of Graphical Virtual Environments by Blind Users via Sensory Substitution. 11, 2 (2016), e0147501. doi:10.1371/journal.pone.0147501
- [52] Metalpop Games. 2025. *UI Accessibility Plugin (UAP) | GUI Tools | Unity Asset Store*. Retrieved December 5th, 2025 from https://assetstore.unity.com/packages/tools/gui/ui-accessibility-plugin-uap-87935?srsltid=AfmBoorDFble5XgdZlh8Oo5a25DAo3FBjwHuDoM056qXOp_7_9t4q7C
- [53] Microsoft. 2025. Microsoft Speech API (SAPI) 5.3. Retrieved May 27, 2025 from [https://learn.microsoft.com/en-us/previous-versions/windows/desktop/ms723627\(v=vs.85\)](https://learn.microsoft.com/en-us/previous-versions/windows/desktop/ms723627(v=vs.85))
- [54] Saikat Mukherjee, I. V. Ramakrishnan, and Michael Kifer. 2003. Semantic bookmarking for non-visual web access. *SIGACCESS Access. Comput.* 77–78 (Sept. 2003), 185–192. doi:10.1145/1029014.1028663
- [55] Ninad Naik. 2024. Probabilistic Consensus through Ensemble Validation: A Framework for LLM Reliability. doi:10.48550/arXiv.2411.06535 arXiv:2411.06535 [cs]
- [56] Vishnu Nair, Jay L. Karp, Samuel Silverman, Mohar Kalra, Hollis Lehv, Faizan Jamil, and Brian A. Smith. 2021. NavStick: Making Video Games Blind-Accessible via the Ability to Look Around. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) (UIST '21). Association for Computing Machinery, New York, NY, USA, 538–551. doi:10.1145/3472749.3474768
- [57] NicholasJJ. 2024. *NicholasJJ/GROMIT*. <https://github.com/NicholasJJ/GROMIT> original-date: 2024-10-19T00:26:46Z.
- [58] NVDA. 2025. *NVDA 2024.4.2 User Guide*. Retrieved May 23, 2025 from <https://download.nvaccess.org/documentation/userGuide.html>
- [59] OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altmenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belugum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Britany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Simón Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Konrad, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nicolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayarvigiya, Chelsea Voss, Carroll Winwright, Justin Jay Wang, Alvin Wang,

- Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL] <https://arxiv.org/abs/2303.08774>
- [60] Achraf Othman, Amira Dhoubi, and Aljazi Nasser Al Jabor. 2023. Fostering websites accessibility: A case study on the use of the Large Language Models ChatGPT for automatic remediation. In *Proceedings of the 16th International Conference on Pervasive Technologies Related to Assistive Environments* (Corfu, Greece) (PETRA '23). Association for Computing Machinery, New York, NY, USA, 707–713. doi:10.1145/3594806.3596542
- [61] Yi-Hao Peng, Dingzeyu Li, Jeffrey P. Bigham, and Amy Pavel. 2025. Morae: Proactively Pausing UI Agents for User Choices. arXiv:2508.21456 [cs.HC] <https://arxiv.org/abs/2508.21456>
- [62] Minoli Perera, Swamy Ananthanarayan, Cagatay Goncu, and Kim Marriott. 2025. The Sky is the Limit: Understanding How Generative AI can Enhance Screen Reader Users' Experience with Productivity Applications. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems* (CHI '25). Association for Computing Machinery, New York, NY, USA, Article 1165, 17 pages. doi:10.1145/3706598.3713634
- [63] Lorenzo Picinali, Amandine Afonso, Michel Denis, and Brian F. G. Katz. 2014. Exploration of architectural spaces by blind people using auditory virtual reality for the construction of spatial knowledge. 72, 4 (2014), 393–407. doi:10.1016/j.ijhcs.2013.12.008
- [64] Playstation 2024. The Last of Us Part II - Accessibility. Retrieved Jan 28, 2024 from <https://www.playstation.com/en-us/games/the-last-of-us-part-ii/accessibility/>
- [65] Alexander Repenning. 2011. Making programming more conversational. In *2011 IEEE Symposium on Visual Languages and Human-Centric Computing* (VL/HCC). 191–194. doi:10.1109/VLHCC.2011.6070398
- [66] Christine Rigden. 1999. 'The Eye of the Beholder'— Designing for Colour-Blind Users. 17 (1999).
- [67] Gary S. Rubin, Mary Feely, Sylvie Perera, Katherin Ekstrom, and Elizabeth Williamson. 2006. The effect of font and line width on reading speed in people with mild to moderate vision loss. *Ophthalmic and Physiological Optics* 26, 6 (2006), 545–554. doi:10.1111/j.1475-1313.2006.00409.x arXiv:<https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1475-1313.2006.00409.x>
- [68] Monika Rychtarikova. 2015. How do blind people perceive sound and soundscape. *Akustika* 23, 1 (2015), 6–9.
- [69] Jeff Sauro and James R. Lewis. 2016. *Quantifying the User Experience, Second Edition: Practical Statistics for User Research* (2nd ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.
- [70] Yoshikazu Seki and Tetsuji Sato. 2011. A Training System of Orientation and Mobility for Blind People Using Acoustic Virtual Reality. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 19, 1 (2011), 95–104. doi:10.1109/TNSRE.2010.2064791
- [71] JooYoung Seo, Sanchita S. Kamath, Aziz Zeidieh, Saairam Venkatesh, and Sean McCurry. 2024. MAIDR Meets AI: Exploring Multimodal LLM-Based Data Visualization Interpretation by and with Blind and Low-Vision Users. In *Proceedings of the 26th International ACM SIGACCESS Conference on Computers and Accessibility* (St. John's, NL, Canada) (ASSETS '24). Association for Computing Machinery, New York, NY, USA, Article 57, 31 pages. doi:10.1145/3663548.3675660
- [72] Hua Shen, Tiffany Kneare, Reshmi Ghosh, Kenan Alkie, Kundan Krishna, Yachuan Liu, Ziqiao Ma, Savvas Petridis, Yi-Hao Peng, Li Qiwei, Sushrita Rakshit, Chenglei Si, Yutong Xie, Jeffrey P. Bigham, Frank Bentley, Joyce Chai, Zachary Lipton, Qiaozhu Mei, Rada Mihalcea, Michael Terry, Diyi Yang, Meredith Ringel Morris, Paul Resnick, and David Jurgens. 2024. Towards Bidirectional Human-AI Alignment: A Systematic Review for Clarifications, Framework, and Future Directions. arXiv:2406.09264 [cs.HC] <https://arxiv.org/abs/2406.09264>
- [73] Andries Smit, Paul Duckworth, Nathan Grinsztajn, Thomas D. Barrett, and Arnu Pretorius. 2024. Should we be going MAD? A Look at Multi-Agent Debate Strategies for LLMs. doi:10.48550/arXiv.2311.17371 arXiv:2311.17371 [cs]
- [74] Chen Sun, Renat Aksitov, Andrey Zhmoginov, Nolan Andrew Miller, Max Vladymyrov, Ulrich Rueckert, Been Kim, and Mark Sandler. 2025. How new data permeates LLM knowledge and how to dilute it. arXiv:2504.09522 [cs.CL] <https://arxiv.org/abs/2504.09522>
- [75] Sarit Felicia Anais Szpiro, Shafeka Hashash, Yuhang Zhao, and Shiri Azenkot. 2016. How People with Low Vision Access Computing Devices: Understanding Challenges and Opportunities. In *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility* (Reno, Nevada, USA) (ASSETS '16). Association for Computing Machinery, New York, NY, USA, 171–180. doi:10.1145/2982142.2982168
- [76] JAIME SÁNCHEZ and MAURICIO LUMBRERAS. 1999. Virtual Environment Interaction Through 3D Audio by Blind Children. 2, 2 (1999), 101–111. doi:10.1089/cpb.1999.2.101 Publisher: Mary Ann Liebert, Inc., publishers.
- [77] Dimitrios Tzavaras, Konstantinos Moustakas, Georgios Nikolakis, and Michael G. Strintzis. 2009. Interactive mixed reality white cane simulation for the training of the blind and the visually impaired. 13, 1 (2009), 51–58. doi:10.1007/s00779-007-0171-2
- [78] Unity Learn 2025. *Unity Learn*. Retrieved September 8th, 2025 from <https://learn.unity.com>
- [79] Jacob T. Urbina, Peter D. Vu, and Michael V. Nguyen. 2025. Disability Ethics and Education in the Age of Artificial Intelligence: Identifying Ability Bias in ChatGPT and Gemini. 106, 1 (2025), 14–19. doi:10.1016/j.apmr.2024.08.014
- [80] Songül Atasavun Uysa and Tülin Düger. 2012. Writing and Reading Training Effects on Font Type and Size Preferences by Students with Low Vision. *Perceptual and Motor Skills* 114, 3 (2012), 837–846. doi:10.2466/15.10.11.24.PMS.114.3.837-846 arXiv:<https://doi.org/10.2466/15.10.11.24.PMS.114.3.837-846> PMID: 22913024
- [81] vellum 2025. *LLM Temperature: How It Works and When You Should Use It*. Retrieved September 5th, 2025 from <https://www.vellum.ai/llm-parameters/temperature>
- [82] VoiceAttack 2025. *VoiceAttack - Voice Recognition for your Games and Apps*. Retrieved Jan 22, 2025 from <https://voiceattack.com/>
- [83] VoiceOver 2025. *Change your VoiceOver settings on iPhone*. Retrieved May 23, 2025 from <https://support.apple.com/en-au/guide/iphfa3d32c50/ios>
- [84] Gareth R. White, Geraldine Fitzpatrick, and Graham McAllister. 2008. Toward accessible 3D virtual environments for the blind and visually impaired. In *Proceedings of the 3rd international conference on Digital Interactive Media in Entertainment and Arts* (New York, NY, USA, 2008-09-10) (DIMEA '08). Association for Computing Machinery, 134–141. doi:10.1145/1413634.1413663
- [85] Lee H. Wurm, Gordon E. Legge, Lisa M. Isenberg, and Andrew Luebker. 1993. Color improves object recognition in normal and low vision. 19, 4 (1993), 899–911. doi:10.1037/0096-1523.19.4.899 Place: US Publisher: American Psychological Association.
- [86] Han Yang, Mingchen Li, Huixue Zhou, Yongkang Xiao, Qian Fang, and Rui Zhang. 2023. One LLM is not Enough: Harnessing the Power of Ensemble Learning for Medical Question Answering. doi:10.1101/2023.12.21.23300380 Pages: 2023.12.21.23300380.
- [87] Yue Yang, Fan-Yun Sun, Luca Weihs, Eli VanderBilt, Alvaro Herrasti, Winson Han, Jiajun Wu, Nick Haber, Ranjay Krishna, Lingjie Liu, Chris Callison-Burch, Mark Yatskar, Aniruddha Kembhavi, and Christopher Clark. 2024. Holodeck: Language Guided Generation of 3D Embodied AI Environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (CVPR). 16227–16237.
- [88] Chandan Yeshwanth, David Rozenberszki, and Angela Dai. 2025. ExCap3D: Expressive 3D Scene Understanding via Object Captioning with Varying Detail. arXiv:2503.17044 [cs.CV] <https://arxiv.org/abs/2503.17044>
- [89] Bei Yuan and Elke Folmer. 2008. Blind here: enabling guitar here for the visually impaired. In *Proceedings of the 10th International ACM SIGACCESS Conference on Computers and Accessibility* (Halifax, Nova Scotia, Canada) (ASSETS '08). Association for Computing Machinery, New York, NY, USA, 169–176. doi:10.1145/1414471.1414503
- [90] Lei Zhang, Jin Pan, Jacob Gettig, Steve Oney, and Anhong Guo. 2024. VRCopilot: Authoring 3D Layouts with Generative AI Models in VR. 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 96, 13 pages. doi:10.1145/3654777.3676451
- [91] Ziyao Zhang, Chong Wang, Yanlin Wang, Ensheng Shi, Yuchi Ma, Wanjun Zhong, Jiachi Chen, Mingzhi Mao, and Zibin Zheng. 2025. LLM Hallucinations in Practical Code Generation: Phenomena, Mechanism, and Mitigation. *Proc. ACM Softw. Eng.* 2, ISSTA, Article ISSTA022 (June 2025), 23 pages. doi:10.1145/3728894
- [92] Zhuohao (Jerry) Zhang, Eldon Schoop, Jeffrey Nichols, Anuj Mahajan, and Amanda Swearngin. 2025. From Interaction to Impact: Towards Safer AI Agent Through Understanding and Evaluating Mobile UI Operation Impacts. In *Proceedings of the 30th International Conference on Intelligent User Interfaces* (IUI '25). Association for Computing Machinery, New York, NY, USA, 727–744. doi:10.1145/3708359.3712153
- [93] Rui Zhao, Hangjie Yuan, Yujie Wei, Shiwei Zhang, Yuchao Gu, Lingmin Ran, Xiang Wang, Zhangjie Wu, Junhao Zhang, Yingya Zhang, and Mike Zheng Shou. 2024. EvolveDirector: Approaching Advanced Text-to-Image Generation with Large Vision-Language Models. In *Advances in Neural Information Processing Systems*, A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang (Eds.), Vol. 37. Curran Associates, Inc., 122104–122129. https://proceedings.neurips.cc/paper_files/paper/2024/file/dcebcd32dfab7c917692c8a9855a351-Paper-Conference.pdf
- [94] Yuhang Zhao, Cynthia L. Bennett, Hrvoje Benko, Edward Cutrell, Christian Holz, Meredith Ringel Morris, and Mike Sinclair. 2018. Enabling People with Visual Impairments to Navigate Virtual Reality with a Haptic and Auditory Cane Simulation. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2018-04-19) (CHI '18). Association for Computing Machinery, 1–14. doi:10.1145/3173574.3173690
- [95] Yuhang Zhao, Ed Cutrell, Christian Holz, Meredith Ringel Morris, Eyal Ofek, and Andrew D. Wilson. 2019. SeeingVR: A Set of Tools to Make Virtual Reality More Accessible to People with Low Vision. <https://www.microsoft.com/en-us/research/publication/seeingvr-a-set-of->

- tools-to-make-virtual-reality-more-accessible-to-people-with-low-vision-2/
- [96] Mingyuan Zhong, Ruolin Chen, Xia Chen, James Fogarty, and Jacob O. Wobbrock. 2025. ScreenAudit: Detecting Screen Reader Accessibility Errors in Mobile Apps Using Large Language Models. In *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems (CHI '25)*. Association for Computing Machinery, New York, NY, USA, Article 1163, 19 pages. doi:10.1145/3706598.3713797

A Appendix

A.1 Appendix 1: Prompt Constructor Instructions

A.1.1 Accessibility Support Instructions: The following is a section about colors: The HEX code represent the color. When asked about the color of an object, answer with natural language color instead of HEX code. Red-green color blindness is a type of color vision deficiency that makes it difficult to distinguish between shades of red and green. To make a scene more accessible for someone with red green color blindness, you should change the color palette. To make a palette for red green colorblind, avoid combining red and green. Also, make sure the new color created are not the same or similar as the other colors in the surroundings. To highlight an object, you can use the GPT Indicator material. To highlight an object without a Renderer, you create a transparent sphere with GPT Indicator material at its location for 5 seconds. To select an object, you can create a transparent sphere with GPT Indicator material at its location for 5 seconds. For simplifying material or texture of an object: When asked to simplify material or texture of an object, create a new material that is closest to the object's original color and assign this new material to the object. If the original color is not provided, use the best guess given the object name. To change the color of an object, first simplify the texture and then change the color.

The following is a section about object and text size: When asked about size of an item, each unit is a meter. Answer how big an object is based on the size in meters. When asked about how big is a text, answer the font size of the text.

The following is a section about spatial relationship between objects: When "me, I, my" is referred, it means the player. When asked about location of objects, answer the object's location relative to the player's location. When asked about the location of one object relative to another, respond by the distance calculated using euclidean distance between the center of the two objects. Be as precise as possible. Also answer how far the item is to the player in common sense, like "the object is close to you" or "the object is far away from you".

The following is a section about scene brightness: To add a light to an area, create a Sphere game object in the area and add a point light to the sphere. To make a light source brighter, adjust the intensity of the Light component.

The following is a section about audio sources: To change the volume of a sound source, change the volume parameter on the AudioSource. To change the pitch of a sound source, change the pitch parameter on the AudioSource. To change the range of a sound source, change the max distance.

When asked to describe the scene or what are in the scene, describe it briefly, group similar objects together instead of listing all items.

A.1.2 Error Prevention Instructions: If the request is general or incomplete, please ask follow-up questions for precise details and contexts, and leave the 'code' field null. If the request says it's not working, please ask follow-up questions to clarify what's happening and suggest users to refine their request. If it's still not working, apologize to users and ask them to try another task. If the request is out of your capability, tell users that the request is out of scope. The types of requests that cannot be achieved include: make zoom/magnifier, edge enhancement, color change on textured materials, object deletion.

A.2 Appendix 2: Scene 1 Demo Example Prompts

A.2.1 Color.

- (1) "What is the color of the cube?" (query)
- (2) "What is the color of the sphere?" (query)
- (3) "Make the color of the cube the same as the sphere." (modification)
- (4) "What is the color of the cube now?" (verification)

A.2.2 Object Size.

- (1) "What is the size of the speaker 1?" (query)
- (2) "Can you make it smaller?" (modification)
- (3) "Will it fit into my hand?" (verification)

A.2.3 Object Location.

- (1) "Grab one of the speakers onto my hand." (modification)
Move around and hear that speaker one is following you
- (2) "What am I grabbing?" (verification)

A.2.4 Scene Brightness.

- (1) "How bright is the scene?" (query)
- (2) "Make the sunlight brighter." (modification)
- (3) "How bright is the scene now?" (verification)

A.2.5 Audio Volume.

- (1) "Mute all speakers" (modification)
- (2) "Unmute speaker one" (modification)
- (3) "Move speaker one much closer to me" (modification)
- (4) "Make the speaker one sound much louder" (modification)

A.2.6 Audio Pitch.

- (1) "Make the pitch of speaker one higher." (modification)
- (2) "Is the pitch of speaker 1 higher than speaker 2 now?" (verification)