Technical Report

IRI-TR-26-01

Institut de Robòtica i Informàtica Industrial

"VisionAssist: An open-source smartphone assistant for AI-based visual accessibility"

Ayse Ozlem Caliskan
Jordi Sanchez Riera

August 2026

CSIC
Consejo Superior de Investigaciones Científicas
UPC UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

VisionAssist: An Open-Source Smartphone Assistant for AI-Based Visual Accessibility

Ayşe Özlem Çalışkan Jordi Sanchez-Riera

Institut de Robòtica i Informàtica Industrial (CSIC-UPC)

Abstract

People with low vision often face challenges in performing everyday tasks that require interpreting visual information. We present **VisionAssist**, an open-source mobile application designed to improve independence by providing AI-powered visual assistance through a smartphone. The application integrates three complementary functionalities within a single interface. First, it enables users to locate specific objects by analyzing the live camera feed. Second, it generates spoken descriptions of captured images, allowing users to identify visual content such as food labels, documents, and everyday objects. Third, it integrates with the smartphone's contacts and calendar to facilitate emergency calls and provide voice-based reminders. The application supports hands-free interaction through voice commands and delivers all feedback using text-to-speech synthesis, making it fully accessible to users with visual impairments. By combining multiple assistive services into a unified platform and releasing the project as open-source software, the proposed solution aims to encourage community contributions and accelerate the development of accessible technologies. The source code is publicly available at: https://github.com/AOzlemC/LowVisionProject.git.

1 Introduction

Visual impairment affects an estimated 285 million people worldwide, substantially limiting their ability to navigate, perceive, and interact with their surroundings independently [Wor23]. Everyday activities such as locating misplaced objects, reading printed text, identifying products, or managing personal schedules remain challenging despite the widespread availability of smartphones. Although mobile devices have become increasingly powerful, general-purpose applications do not adequately address the accessibility needs of blind and low-vision (BLV) users. Consequently, there is a growing demand for intelligent assistive technologies that enable natural and independent interaction with the physical and digital environment.

Recent advances in artificial intelligence, particularly the emergence of Vision-Language Models (VLMs) and Multimodal Large Language Models (MLLMs), have transformed the capabilities of mobile assistive systems. By jointly reasoning over visual and textual information and supporting natural-language interaction, these models enable users to obtain scene descriptions, ask follow-up questions, recognize objects, read printed text, and retrieve contextual information from images. Their conversational interface makes them especially well suited for BLV users, allowing visual perception to be replaced by spoken dialogue in a natural and intuitive manner.

At the same time, significant progress in model compression, efficient architectures, and mobile hardware acceleration has made it increasingly feasible to deploy powerful multimodal models on resource-constrained devices such as smartphones. This has led to the rapid development of AI-powered assistive applications that address a variety of individual tasks, including scene description, optical character recognition (OCR), object recognition, navigation assistance, and obstacle detection [fin24, Goo23, Mic23, Env24, For24, MIP24, Nav24, Voi24]. Other services, such as Be My Eyes [Be 23] or Aira [Air24], complement AI with remote human assistance, enabling users to obtain reliable answers for complex situations.

Despite these advances, current assistive solutions remain fragmented. Most applications focus on a single functionality or a limited set of related tasks, requiring users to switch between multiple applications depending on their immediate needs. Furthermore, many state-of-the-art commercial systems

rely on proprietary cloud-based models and closed-source software, limiting transparency, extensibility, reproducibility, and personalization. Although recent MLLM-based assistants have significantly improved conversational scene understanding, they generally provide holistic descriptions of the environment rather than supporting targeted, query-driven object localization (e.g., "Where are my keys?"). Similarly, personal productivity functions such as calendar management or communication are typically handled by separate voice assistants, resulting in a fragmented user experience.

To address these limitations, we present a unified, cross-platform assistive mobile application that integrates multimodal visual perception with personal assistant capabilities within a single extensible framework. The proposed system provides three complementary interaction modes. The Find mode enables users to locate specific objects in their surroundings through natural-language queries. The Describe mode supports scene understanding, image description, and text reading using multimodal AI models. Finally, the Personal Assistant mode allows users to perform everyday organizational tasks, including calendar consultation and phone calls, through voice interaction. All interactions are speech-first, allowing users to operate the application hands-free without requiring visual attention to the screen, while responses are delivered through text-to-speech synthesis.

The main contributions of this work are threefold:

- A unified mobile assistive framework that combines object localization, scene description, optical character recognition (OCR), contact calling, and calendar assistance into a single speech-driven application.
- A carefully designed speech-driven user interface that enables users to interact with the application through voice commands, simplifying the interaction process and improving accessibility.
- An open-source, modular implementation that promotes reproducibility, community engagement, and the continued development of new assistive functionalities.

2 Related Work

Despite the significant advances in Vision–Language Models (VLMs) and, more recently, Multimodal Large Language Models (MLLMs), these technologies cannot yet be directly deployed on resource-constrained mobile devices without important trade-offs. Although recent works have demonstrated remarkable performance in multimodal reasoning and visual understanding [ZHJL24, SQTA+25, ZFX+25, bla24, LZS+24], their computational and memory requirements often exceed the capabilities of current smartphones. As a result, fully on-device inference typically leads to excessive memory consumption, unstable execution, or long response times, making real-time assistive applications impractical. Consequently, most existing mobile assistive applications rely either on lightweight model variants or on cloud-based inference to provide acceptable performance.

Table 1 summarizes the main characteristics of several representative mobile assistive applications. As shown, existing solutions can be broadly categorized according to their primary functionalities and deployment strategies. Some applications provide ´´human assistance´´, connecting users with a remote volunteer or trained agent to interpret visual information. Others are limited to a single mobile platform, supporting either Android or iOS but not both. Applications also differ in their execution model, with some operating entirely on the device while others rely on cloud-based processing. In addition, only a subset of applications offers navigation capabilities.

Compared with these solutions, the proposed application combines multiple AI-powered vision functionalities —including object localization, scene description, and optical character recognition (OCR)— within a unified cross-platform interface. Furthermore, it extends the functionality of conventional visual assistance applications by incorporating voice-controlled personal assistant features, such as contact calling and calendar event retrieval. To the best of our knowledge, the integration of these smartphone assistance capabilities into the same speech-driven application has not been provided by existing mobile assistive solutions.

3 Application Workflow

The application is developed using a modular architecture, allowing it to be easily extended with new functionalities in the future while also facilitating improvements to existing features. The workflow is

Table 1: Comparison of existing mobile assistive applications for people with visual impairments. Object denotes object localization, Scene denotes scene description, OCR refers to optical character recognition, Nav. indicates navigation support, and Human represents remote human assistance.

Application	Object	Scene	OCR	Nav	Human	Calls	Calendar	Offline	Platform
Aira	✓	✓	✓	✓	✓	✗	✗	✗	iOS, Android
Be My Eyes	✓	✓	✓	Partial	✓	✗	✗	✗	iOS, Android
Find My Things	✓	✗	✗	✗	✗	✗	✗	✓	iOS
Google Lookout	✓	✓	✓	Partial	✗	✗	✗	Partial	Android
Seeing AI	✓	✓	✓	Partial	✗	✗	✗	Partial	iOS
Envision AI	✓	✓	✓	Partial	✗	✗	✗	Partial	iOS, Android
Sullivan+	✓	✓	✓	Partial	✗	✗	✗	Partial	iOS, Android
BlindSquare	✗	✗	✗	✓	✗	✗	✗	Partial	iOS
NaviLens	Marker	✗	Limited	✓	✗	✗	✗	✓	iOS, Android
Voice Dream Scanner	✗	✗	✓	✗	✗	✗	✗	✓	iOS
VisionAssist (ours)	✓	✓	✓	✗	✗	✓	✓	Partial	iOS, Android

designed to be intuitive and user-friendly, consisting of a sequence of well-defined stages, as illustrated in Figure 1.

When the user launches the application, the initial screen captures both an image from the device's camera and the user's voice command. The user is then presented with a set of selectable options. Specifically, the application provides two operating modes: the **AI Vision** mode and the **Personal Assistant** mode. Once the desired mode and its corresponding options have been selected, the collected data are passed to a lightweight multimodal large language model (MLLM), which generates a textual response. Finally, the generated text is converted into speech through a text-to-speech (TTS) module, providing the user with an audible response.

3.1 AI Vision

The AI Vision module is designed to support three computer vision tasks: *object localization*, *scene description*, and *optical character recognition (OCR)*. For each task, the application captures an image from the smartphone's camera and combines it with a task-specific predefined prompt. The captured image and the corresponding prompt are then transmitted to a cloud-based AI module, which processes the multimodal input and generates an appropriate textual response. Offloading inference to the cloud enables the use of a better Vision–Language Model while avoiding the computational and memory limitations of mobile devices. The generated response is subsequently converted into speech through the text-to-speech (TTS) module, enabling hands-free interaction for the user.

Object localization. In this task, the user specifies a target object via a voice command (e.g., "find my keys"), which a speech recognition module transcribes into text. To ensure system robustness and prevent silent failures, the application immediately validates this input; if no object is specified, it issues a spoken error message prompting the user to try again. Once validated, the transcribed object name and a captured image are forwarded to the AI module with a predefined prompt instructing it to locate the item within two sentences. The model analyzes the scene and generates a concise spatial description of the object's location (e.g., "top-right of the frame"), which a text-to-speech (TTS) module then reads aloud to confirm the result to the user.

> *Prompt: "Find [object] in this image. Tell me if it's visible and where it is (e.g., center, top-left, bottom-right). Be brief, max 2 sentences."*

Scene description. In this task, the image captured by the smartphone's camera is directly passed to the AI module, which generates a natural language description of the observed scene. Since large language models (LLMs) often produce lengthy responses that may include unnecessary details, the prompt is carefully designed to generate concise and informative descriptions. Specifically, it imposes response length constraints to avoid verbose output, making the generated description more suitable for a listening-based interface. The resulting text is then converted into speech using a text-to-speech (TTS) synthesizer, enabling the user to receive the scene description audibly.

> *Prompt: "Describe this scene for a blind person in 2–3 short sentences. Be concise."*

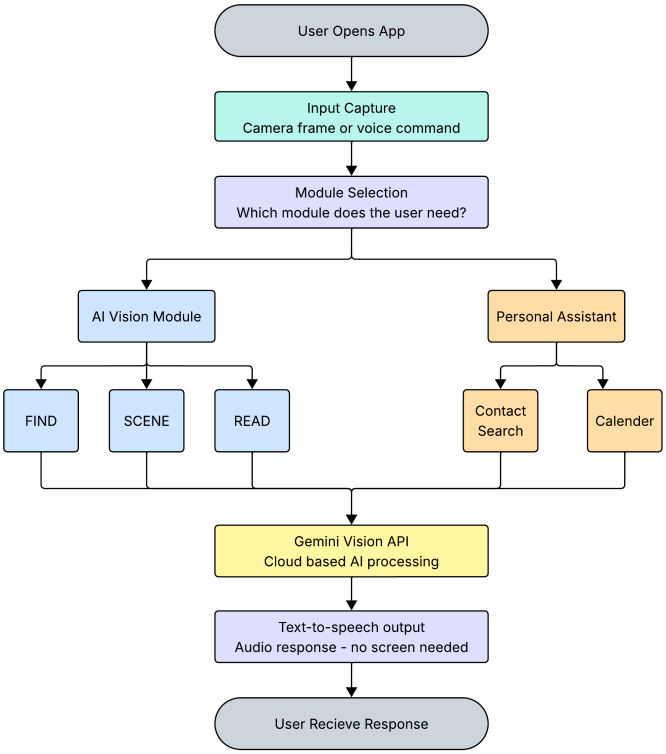

Figure 1: General workflow of the mobile intelligent object localization assistant. User input is captured via camera frame or voice command, routed through the AI Vision Module (FIND, SCENE, READ) or Personal Assistant (Contact Search, Calendar), processed by the Gemini Vision API, and returned as Text-to-Speech audio output.

Optical character recognition (OCR). In this task, the image captured by the smartphone's camera is expected to contain readable text, such as a product label, street sign, restaurant menu, or any other printed document with recognizable characters. The captured image is then sent to the AI module, which performs optical character recognition (OCR) to extract the text. The recognized text is subsequently converted into speech using a text-to-speech (TTS) module, enabling the user to hear the extracted information.

Prompt: "Extract and read all text in this image clearly."

3.2 Personal Assistant

The Personal Assistant module enables users to access common smartphone functionalities, such as making phone calls and managing their personal agenda, entirely through voice commands. By eliminating the need for manual interaction, the module provides accessible, voice-driven quick actions for frequently performed daily tasks. Each successfully executed action is immediately followed by a text-to-speech (TTS) confirmation, ensuring continuous auditory feedback and maintaining the speech-first interaction paradigm throughout the application. As a result, users can complete all supported tasks without needing to look at or interact with the screen. Unlike the AI Vision module, the personal assistant module works entirely on the device and does not require an internet connection, ensuring reliable access to calendar information even when offline.

Phone calls. In this mode, the user can initiate a phone call by specifying a contact's name through either text or voice input. Voice commands are first transcribed into text and then matched against the device's native contact list using the Expo Contacts API. Once a matching contact is found, the application initiates the phone call automatically. The module supports natural language commands such as *"Call Mom"*, *"Ring John"* and *"Phone [name]"*, providing a flexible and intuitive user

experience. Throughout the interaction, the application delivers text-to-speech (TTS) confirmations to inform the user of each step, ensuring continuous auditory feedback.

Calendar Events. In this mode, the assistant retrieves events from the device's native calendar using the Expo Calendar API and presents them to the user through text-to-speech (TTS), ordered chronologically. The module supports natural language queries such as *"Any meetings today?"*, *"Check tomorrow's schedule"* and *"What's happening this week?"*. Based on the user's request, the application automatically infers the appropriate time interval (e.g., the current day, the next day, or the following week) and retrieves the corresponding events.

4 Implementation and Design

The application is implemented following the workflow illustrated in Figure 1. The user interface, including all screens and interactive components, is developed using React Native together with the Expo framework [Exp24]. This technology stack was selected for its cross-platform capabilities, enabling the application to run on both Android and iOS devices from a single codebase.

Figure 2 presents screenshots of the main application interfaces. Figure 2(a) shows the home screen, from which the user can choose between the AI Vision module and the Personal Assistant module. Figures 2(b)–(d) illustrate the three AI Vision modes (Find, Scene, and Read), while Figure 2(e) shows the Personal Assistant interface.

The AI Vision functionalities are implemented using the Gemini Vision API [Goo24], specifically the `gemini-2.5-flash` model, which provides multimodal image understanding while maintaining low response latency. The generated textual responses are converted into speech using a text-to-speech (TTS) engine configured with a speech rate of (0.9×) the default speed, a value empirically selected to improve intelligibility and listening comfort for users.

(a) (b) (c) (d)

Figure 2: Application interface screenshots. (a) Example of output for the Object recognition mode where the query corresponds to the spoken word "Bottle"; (b) Example of Scene description mode where camera captures an indoor living room and the model returns a spoken 2–3 sentence description; (c) OCR output on a visible text extracted from the camera frame; (d) Personal Assistant Module presenting the available voice-controlled functionalities, including contact calling and calendar event retrieval.

5 Experiments

Several experiments were conducted to evaluate the performance and effectiveness of the proposed application. First, multiple AI models were compared to identify the most suitable model for the vision-based tasks. After selecting the best-performing model, its performance was evaluated across the three AI Vision functionalities: object localization, scene description, and optical character recognition (OCR). In addition, the functionality and responsiveness of the Personal Assistant module were assessed through a series of task-oriented evaluations. The following sections describe the experimental setup, methodology, and results for each of these evaluations in detail.

5.1 AI Backbone Selection

Several state-of-the-art Vision–Language Models (VLMs) were subsequently evaluated for on-device deployment. Qwen2-VL [BBY+23] was initially considered because of its strong multimodal capabilities; however, compatibility limitations prevented stable deployment on the target mobile platform. LLaVA [LLWL23] received the most extensive evaluation due to its promising performance reported in the literature. Although the model successfully completed one or two inference cycles, it consistently crashed under sustained execution because of out-of-memory (OOM) conditions on the 8,GB device, making it unsuitable for continuous operation. Florence [YCC+21] was also evaluated experimentally but exhibited computational and memory requirements that exceeded the practical capabilities of the target hardware. Finally, Moondream [Moo24], despite its compact architecture and apparent suitability for edge deployment, repeatedly failed during inference by returning the generic response "I couldn't identify the image," preventing reliable execution.

Overall, the experimental evaluation demonstrated that none of the assessed on-device Vision–Language Models (VLMs) provided the level of stability, reliability, and performance required for the proposed application under the available hardware constraints. Consequently, the Gemini Vision API was selected as the final AI backend. Although this cloud-based solution requires an internet connection, it consistently produced accurate responses with acceptable latency and substantially greater robustness than the evaluated on-device alternatives. Under standard Wi-Fi connectivity, the Gemini API achieved an average response latency of 2–4 seconds, which was considered acceptable for the intended assistive application. Since the system relies exclusively on the pretrained Gemini model for all vision tasks, no task-specific dataset was collected and no additional training or fine-tuning was performed. A quantitative comparison of the evaluated models is presented in Table 2.

Table 2: Comparison of evaluated models for mobile deployment.

Model	**Deployment**	**Observed Behaviour**
Qwen2-VL [BBY+23]	On-device	Limited practical success; mobile compatibility constraints.
LLaVA [LLWL23]	On-device	1–2 successful inferences; out-of-memory (OOM) crashes under RAM pressure.
Florence [YCC+21]	On-device	Insufficient practical performance on target device.
Moondream [Moo24]	On-device	Repeatedly returned "I couldn't identify the image"; unreliable inference despite compact size.
Gemini [Goo24]	Cloud	Accurate and responsive; latency increased over sustained use; connectivity-dependent.

5.2 Application Validation

The proposed application was validated in two representative indoor environments, namely a home and an office, under typical indoor lighting conditions. Each functionality was evaluated independently using multiple images and user queries representative of real-world usage scenarios. The evaluation

was conducted separately for each of the application's operating modes to assess its performance, reliability, and overall functionality under consistent testing conditions.

5.3 AI Vision

Object Localization

The object localization functionality was evaluated using 10 commonly encountered household and office objects. For each object, five images were captured from different viewpoints and orientations, resulting in a diverse test set representative of everyday usage conditions. The AI model was assessed based on its ability to correctly identify the requested object and report its approximate location within the scene.

Most objects were detected reliably across the evaluated images. The phone and remote control achieved 100% detection accuracy, as their distinctive visual characteristics enabled consistent recognition under all tested conditions. The majority of recognition failures occurred in cluttered scenes or under low-light conditions, where object visibility and image quality were reduced. The per-object recognition accuracy is summarized in Table 3.

Table 3: Object recognition accuracy per object.

	Mouse	**Pen**	**Cup**	**Keyboard**	**Bottle**	**Phone**	**Keys**	**Book**	**Glasses**	**Remote**	**Overall**
Accuracy	80%	80%	80%	80%	80%	100%	80%	80%	80%	100%	**84%**

Scene Description

The scene description functionality was evaluated using a dataset of 10 images depicting a variety of everyday indoor scenarios. For each image, the AI model generated a concise description consisting of two to three sentences. Across all test cases, the responses were informative, coherent, and consistently adhered to the prompt-imposed length constraints. This controlled response length made the generated descriptions well suited for direct delivery through the text-to-speech (TTS) module, ensuring a natural and efficient listening experience. An example of a generated scene description is presented below.

> **Input:** Living room camera frame (sofa, TV, cabinet, cat).
> **Output:** *"This indoor scene features a light grey sofa on the left, and a large, wall-mounted TV above a decorative white cabinet on the right. A black and white cat is lounging on the wooden floor near an open door."*

Optical character recognition (OCR)

Similarly, the optical character recognition (OCR) functionality was evaluated using 10 images containing a variety of printed text sources, including product labels, signs, and documents. The AI model successfully extracted the text in 9 out of the 10 test images, achieving an overall accuracy of 90%. The only failure occurred for an image containing a stylized handwritten label, which proved challenging for the OCR system. An example of a successful OCR result is presented below.

> **Input:** Product label (shampoo bottle).
> **Output:** *"Head & Shoulders Classic Clean Shampoo. 400 ml. For normal hair. Directions: Apply to wet hair, lather, rinse. Repeat if necessary."*

5.4 Personal Assistant

Contact Search

To evaluate the contact access functionality of the Personal Assistant module, we conducted a series of experiments in which users initiated phone calls using natural voice commands. The evaluated queries, listed in Table 4, included a variety of contact names and natural language expressions. A total of 10 voice queries were tested, and the corresponding results are summarized in Table 4.

The application successfully matched and retrieved contacts when the spoken query exactly matched or partially corresponded to the stored contact name. Most failures occurred when the voice command was overly informal or did not contain a substring that could be associated with any contact in the device's contact list. Overall, the results demonstrate that the contact retrieval mechanism performs reliably for common voice commands while highlighting the importance of recognizable name matching for successful call initiation.

Table 4: Contact Search results. Ok = correct contact found and call initiated.

Query	**Stored Name**	**Result**	**Note**
"Call mom"	Mom	Ok	Exact match
"Call dad"	Dad	Ok	Exact match
"Call John"	John Smith	Ok	Partial match
"Call Sarah"	Sarah Johnson	Ok	Partial match
"Call Dr. Ahmed"	Dr. Ahmed Yilmaz	Ok	Partial match
"Call Emma"	Emma Clarke	Ok	Partial match
"Call Alex"	Alex	Ok	Exact match
"Call Olivia"	Olivia Brown	Ok	Partial match
"Call Lisa"	Lisa Taylor	×	No match found
"Call the doctor"	Dr. Ahmed Yilmaz	×	Descriptive query unsupported
Accuracy		**8/10 (80%)**	

Calendar Events

Similarly, the calendar event retrieval functionality was evaluated using 10 representative voice queries covering events scheduled for the current day, the following day, and the upcoming week. The corresponding queries and results are summarized in Table 5.

The Personal Assistant module achieved 100% accuracy across all test cases. Since this functionality operates entirely on the device through the native Expo Calendar API, its performance is independent of AI model variability and network connectivity. For every query, the application successfully retrieved the relevant events and presented them in chronological order, correctly reporting the event title, date, and time through the text-to-speech (TTS) module. These results demonstrate the reliability of the on-device calendar integration for voice-based personal assistance.

Table 5: Calendar Events results. Ok = correct events retrieved and read aloud.

Query	**Look-ahead**	**Events**	**Result**
"Any meetings today?"	1 day	2	Ok
"Check today's schedule"	1 day	0	Ok (correctly reported none)
"Check tomorrow's schedule"	2 days	1	Ok
"Any events tomorrow?"	2 days	3	Ok
"What's happening this week?"	7 days	5	Ok
"Any meetings this week?"	7 days	5	Ok
"Do I have anything today?"	1 day	1	Ok
"What's on my calendar?"	1 day	2	Ok
"Check my schedule"	1 day	0	Ok (correctly reported none)
"Upcoming events"	7 days	4	Ok
Accuracy			**10/10 (100%)**

6 Conclusions

This work presents a mobile assistive application that combines AI-powered vision capabilities with voice-based smartphone interaction to address key limitations of existing visual assistance tools. A systematic evaluation of several on-device Vision–Language Models (VLMs) highlighted the challenges of deploying multimodal AI on resource-constrained devices and motivated the adoption of a cloud-based vision pipeline. The resulting hybrid architecture provides a practical balance between performance, reliability, and accessibility.

The final application offers five core functionalities: object localization, scene description, optical character recognition (OCR), contact search, and calendar event retrieval. All features are accessible through a speech-first interface, enabling users to perform tasks without relying on visual interaction with the screen. Moreover, the cross-platform implementation based on React Native and the Expo framework provides a modular and extensible foundation for future research in mobile accessibility.

The main limitation of the proposed system is its reliance on an internet connection for the AI Vision module. Future work will focus on integrating lightweight on-device Vision–Language Models as mobile AI technology matures, conducting user studies with visually impaired participants, and performing a more comprehensive quantitative evaluation of the system's performance.

References

[Air24] Aira Tech Corp. Aira: On-demand visual interpretation service, 2024.

[BBY+23] Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*, 2023.

[Be 23] Be My Eyes. Be my ai, 2023.

[bla24] Blave, 2024.

[Env24] Envision Technologies. Envision ai: App for blind and visually impaired users, 2024.

[Exp24] Expo. Expo documentation, 2024.

[fin24] Findmythings, 2024.

[For24] Forea Inc. Sullivan+: Ai-based vision assistance app, 2024.

[Goo23] Google. Lookout, 2023.

[Goo24] Google DeepMind. Gemini api documentation, 2024.

[LLWL23] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[LZS+24] Ruiping Liu, Jiaming Zhang, Angela Schön, Karin Müller, Junwei Zheng, Kailun Yang, Anhong Guo, Kathrin Gerling, and Rainer Stiefelhagen. Objectfinder: An open-vocabulary assistive system for interactive object search by blind people. *arXiv preprint arXiv:2412.03118*, 2024.

[Mic23] Microsoft. Seeing ai, 2023.

[MIP24] MIPSoft. Blindsquare: Gps app for the blind and visually impaired, 2024.

[Moo24] Moondream AI. Moondream: A tiny vision-language model, 2024.

[Nav24] NaviLens. Navilens: Assistive navigation system for blind users, 2024.

[SQTA+25] Ajay Narayanan Sridhar, Fuli Qiao, Nelson Daniel Troncoso Aldas, Yanpei Shi, Mehrdad Mahdavi, Laurent Itti, and Vijaykrishnan Narayanan. Navisense: A multimodal assistive mobile application for object retrieval by persons with visual impairment. In *Proceedings of the 27th International ACM SIGACCESS Conference on Computers and Accessibility*, 2025.

[Voi24] Voice Dream. Voice dream scanner: Ocr and text-to-speech app, 2024.

[Wor23] World Health Organization. Blindness and vision impairment, 2023.

[YCC+21] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, Ce Liu, Mengchen Liu, Zicheng Liu, Yumao Lu, Yu Shi, Lijuan Wang, Jianfeng Wang, Bin Xiao, Zhen Xiao, Jianwei Yang, Michael Zeng, Luowei Zhou, and Pengchuan Zhang. Florence: A new foundation model for computer vision, 2021.

[ZFX+25] He Zhang, Nicholas J. Falletta, Jingyi Xie, Rui Yu, Sooyeon Lee, Syed Masum Billah, and John M. Carroll. Enhancing the travel experience for people with visual impairments through multimodal interaction: Navigpt, a real-time ai-driven mobile navigation system. In *Companion Proceedings of the 2025 ACM International Conference on Supporting Group Work*, page 29–35, 2025.

[ZHJL24] Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. Vision-language models for vision tasks: A survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, 46(8):5625–5644, 2024.