Plan for Vision AI Glasses to Assist the Blind

Zachary Carson

July 30th, 2025

Abstract

Often, those suffering with vision problems do not have the resources to navigate their environment, let alone know everything that's happening around them. While there are already resources to help the blind, many of these resources do not cover the entire issue or take significant time and resources to deploy. Fortunately, however, rapid progression in technology within the last 5 years has allowed more possibilities to alleviate certain issues that the blind face.

The objective of this paper is to plan out one possible solution: a pair of camera glasses that uses A.I. object detection and spatial computing to identify and locate objects, feeding the positional information to an LLM to generate a response back through audio. Finding data to reach this conclusion is done primarily through research into previous iterations of similar designs. The key findings of this paper include the specific hardware and software requirements needed to achieve this design, as well as benefits and possible considerations/drawbacks.

Introduction

Any degree of visual impairment will cause significant difficulty in a person's day-to-day life. In many situations, they must rely heavily on the environment around them, whether it's crosswalks that have audio cues when they can go, or the sounds of traffic and footsteps. They might also have resources of their own, like a cane or guide dog that they can use to further detect obstacles.

Regardless, all of these methods, even when used in tandem, can't be used consistently or require significant time or money investment. For example, intersection audio cues aren't going to be found in every town or city, especially if the area is not very developed. Additionally, lots of cars, notably electric vehicles, are being manufactured to produce much less noise than before, making it difficult to rely on noise. Or what if the person doesn't have enough resources to afford and take care of a guide dog? There are many situations in which visual impairment can pose a significant challenge for people who don't have adequate access to resources.

The objective of this study is to find a way to best leverage the latest developments in hardware and artificial intelligence in order to further expand the resources that blind people have. The design that will be explored is a camera glasses device that allows a user to ask about their surroundings and receive that information back through text-to-speech. This study holds significance in that it has the potential to set a new benchmark in technology focused on visual impairment.

The scope of this study uses the research and experiments of studies and other sources to draw this conclusion. Although the best effort will be made to identify potential strengths and drawbacks, actual performance isn't able to be determined until a live prototype is created.

The next sections in this paper are as follows: 1.) a literature review of previous studies/projects, 2.) the methodologies used for research and data collection, 3.) the results gathered from the study, and 4.) an interpretation and analysis of the key findings.

Literature Review

Although a fairly new concept, the concept of AI vision glasses for the blind has already been widely explored in studies and has been brought into fruition as working prototypes or even commercial products.

A study by the National Library of Medicine goes into two of these commercial products, the Ray-Ban Meta and Envision Glasses. Some of their features include high-definition cameras, an AI powered assistant, and AR/VR capabilities. Envision specifically focuses on relaying visual information back to the user via GPT integration, "allowing users to ask the glasses specific questions, like to summarize text, or only reading vegan items from a menu". While this sounds like an ideal product, it is only affordable for those willing to pay upwards of $3,499.

Meta glasses share a similar function, while having their own advanced features, allowing users to "ask Meta AI questions about what they are looking at . . . receive auditory information about their environment, read text aloud, recognize faces, or get directions" (Waisberg, 2024).

Overall, these examples represent a wider movement from large companies to provide assistive technology to those with visual impairments. Additional research into this technology helps to showcase DIY projects that can have similar functionality and capabilities. One report from the Indonesian Journal of Electrical Engineering and Computer Science highlights a camera device that is connected to a Raspberry Pi, and even uses a sonar sensor to detect when objects are close to the user. Much like the ones on the market, it also uses AI to recognize objects, read messages, and provide voice commands back to the user (Gollagi, 2023).

However, a somewhat significant research gap also emerges, as there are currently very few studies that test the effectiveness or user satisfaction of a device with these specifications. Regardless, the amount of research and testing of this sort of device within just the past two years, especially by large companies such as Meta, is a strong predictor of its success.

Knowing what has been done before, the best way to differentiate this project will be to produce a highly effective and portable product, while also maintaining a low enough budget that it could be sold at a relatively affordable price.

Methodology

To assess the feasibility of an AI-powered smart glasses prototype, a positivist research approach was used for creating the design and architecture of the device. Considering the scope and focus of this study, the data collected is mostly secondary research, focusing on what components would be required for the device, and how they would properly contribute to its overall design and function. This was achieved through analyzing existing devices and software, evaluating their performance and features, and designing a full prototype using those components.

4.1: Functional Requirements

The following functions were identified for the device, in order to define the scope of and choose the necessary components. These functions would support the main use cases of real-time object recognition, contextual guidance, and audio processing/feedback needed for the prototype.

1. **Stereo depth estimation** for localizing objects in 3D space. Distance coordinates
2. **Real-time object detection**, using a lightweight, edge-compatible model. Provides a name for the object.
3. **Voice-based user input**, using speech-to-text in order to transcribe the audio.
4. **Cloud communication** for sending visual data and voice input to a GPT model for interpretation, and then receiving a response from the model
5. **Audio feedback** to the user using on-device TTS functionality

4.2: Component Identification

The general components that were identified and researched are as follows:

1. A **stereo camera:** locate where objects are. This is done using binocular disparity, only possible when having two lenses.
2. **Object detection algorithm**: identify what the object is (cat, dog, person, car, etc.). This will run on the stereo camera as well.
3. **Smartphone application**: collects data from glasses, allows user to ask a question about their surroundings, and transcribes it to text.
4. **Cloud-based GPT model** connects to smartphone application, processes user request and object data, returns a response.

This choice of components was decided based on performance, cost, compatibility, ergonomics, energy consumption, and portability.

4.3: Component Selection Criteria

This device was designed to achieve the best possible balance between:

1. **Portability**
2. **Affordability**
3. **Power efficiency/consumption**
4. **Real-time performance**
5. **Cost**

Doing extensive research into other options, such as using pre-built glasses, a Raspberry Pi, various microcontrollers, or a Jetson Nano as edge devices, a device using cloud-computing was found to have the best balance between all these different aspects.

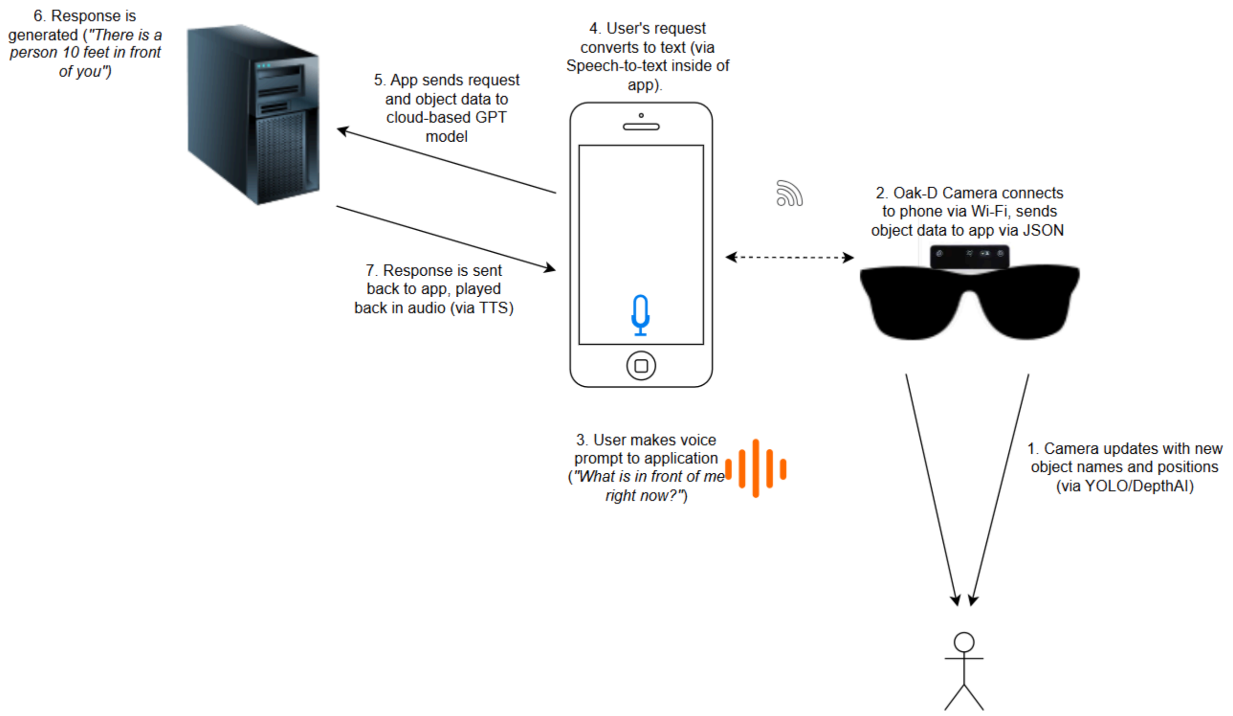

6. Response is generated ("There is a person 10 feet in front of you")
5. App sends request and object data to cloud-based GPT model
4. User's request converts to text (via Speech-to-text inside of app).
2. Oak-D Camera connects to phone via Wi-Fi, sends object data to app via JSON
7. Response is sent back to app, played back in audio (via TTS)
3. User makes voice prompt to application ("What is in front of me right now?")
1. Camera updates with new object names and positions (via YOLO/DepthAI)

Figure 1, System Architecture Diagram

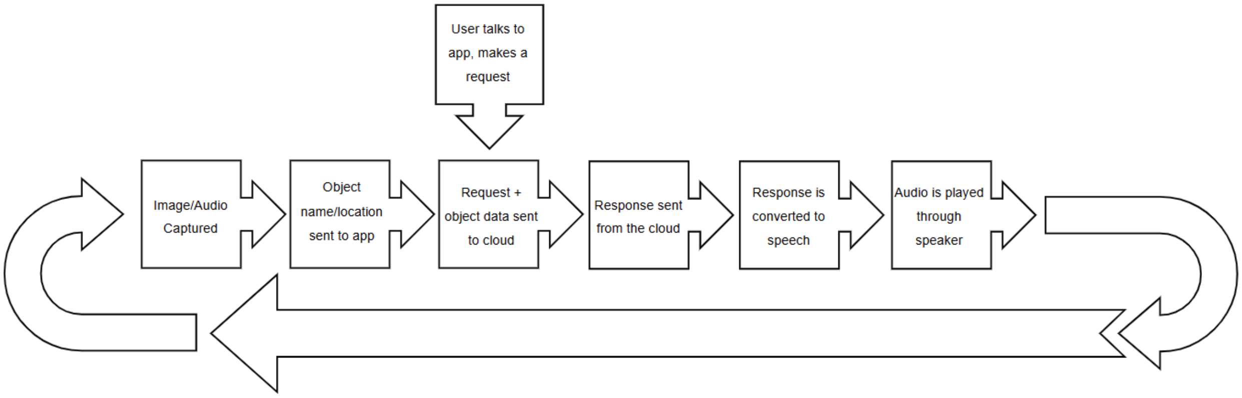

User talks to app, makes a request
Image/Audio Captured
Object name/location sent to app
Request + object data sent to cloud
Response sent from the cloud
Response is converted to speech
Audio is played through speaker

Figure 2, Data Flow Diagram

4.4: Justification of Tools

- **React Native** was selected for mobile development to ensure **cross-platform compatibility** and built-in APIs.
- **YOLOv8** was selected for its lightweight real-time detection capabilities.
- **OAK-D IoT** was selected for its powerful VPU which can run both DepthAI object localization and object detection models like YOLO. It can also connect to iOS and Android smartphones via a Wi-Fi access point, allowing cross-platform functionality.
- **OpenAI's ChatGPT models** were chosen for their flexible natural language understanding and low cost per request.

Results & Analysis

5.1: Stereo Camera

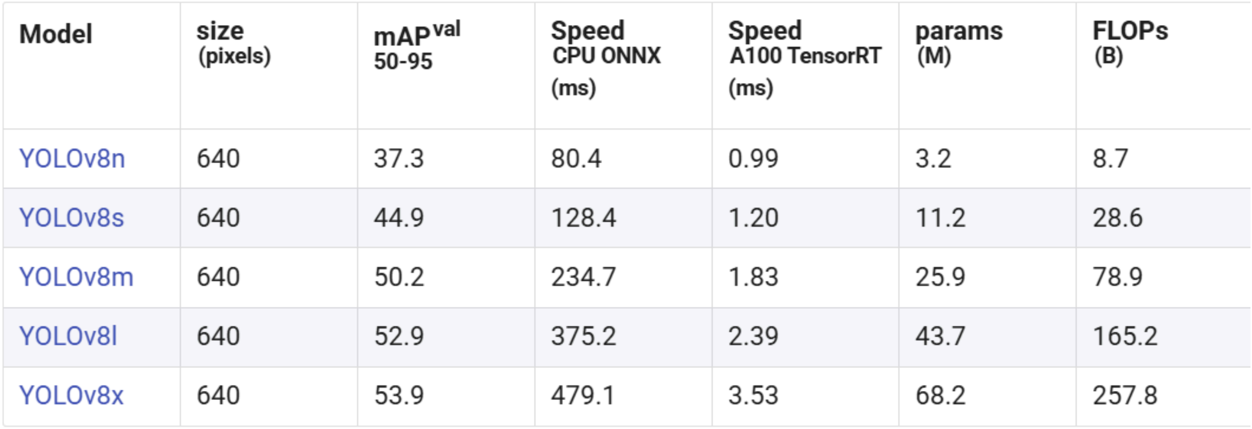
For the camera, the OAK-D IoT was considered the best choice for the overall product. Weighing 90 grams, it is lightweight and quite easy to attach to a glasses frame. However, the special part of the OAK-D device is that it hosts a Myriad X VPU (Visual Processing Unit) chip, which is specialized for machine learning and computer vision algorithms. Specifically, this device hosts DepthAI, which allows objects to be properly tracked and have their 3D coordinates calculated. Additionally, Oak-D IoT can host object detection models within DepthAI, which detects and properly labels them (Kumaran, 2024). The main benefit of a VPU is that it allows these complex vision algorithms to be locally hosted on the camera, which reduces dependency on external processing and makes the entire device less complex (Luxonis, 2025).

5.2: Object Detection A.I.

The second most important aspect of this device is the AI Object Detection model that will be hosted on DepthAI/Oak-D. One consideration that must be made with this is inference time, in other words the time that it takes for the algorithm to detect an object and label it. Obviously, the lower the inference time, the closer it is to operating in real-time, which is ideal in this scenario as you want the device to be able to pick up on new objects quickly.

Another aspect of this device to consider is the accuracy of the algorithm. There are multiple ways to measure accuracy in this instance, however the most common way is with mean average precision (mAP). The mAP is graded on a scale of 0-1, with 1 being the best possible accuracy.

The YOLO ("You Only Look Once") object detection algorithm, which is considered state-of-the-art in its kind, was seen as the software to pair with the device. A major strength of YOLO is that there are many different versions of a model depending on the processing power of the device running it, allowing it to accommodate smaller, more portable devices. As of right now, YOLO8_s (“small” model) will be running on the device, as it remains comparable to its larger counterparts, while still being able run on an Oak-D device (Roboflow, 2023).



Model	size (pixels)	mAP^{val} 50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	params (M)	FLOPs (B)
YOLOv8n	640	37.3	80.4	0.99	3.2	8.7
YOLOv8s	640	44.9	128.4	1.20	11.2	28.6
YOLOv8m	640	50.2	234.7	1.83	25.9	78.9
YOLOv8l	640	52.9	375.2	2.39	43.7	165.2
YOLOv8x	640	53.9	479.1	3.53	68.2	257.8

Figure 3, YOLOv8 model performance

5.3: Smartphone application

After the Oak-D IoT labels and localizes an object, it will send that information to the smartphone application. However, this will be done wirelessly (through Wi-Fi), as opposed to a USB cable. The first reason for this is simply user ergonomics; it isn’t comfortable or practical to have a wire running from the camera glasses to one’s phone.

The second reason is that Wi-Fi connection allows the data from the camera to be sent to both iOS and Android devices. If it were done through USB connection, the smartphone device would require a DepthAI SDK, which is exclusive to Android. With wireless connection, both devices need to be connected to a network, and Oak-D will send the information via JSON, able to be read by iOS and Android devices alike. JSON is also a very standard data format, effective yet lightweight enough that it can easily be transmitted without excessive bandwidth usage.

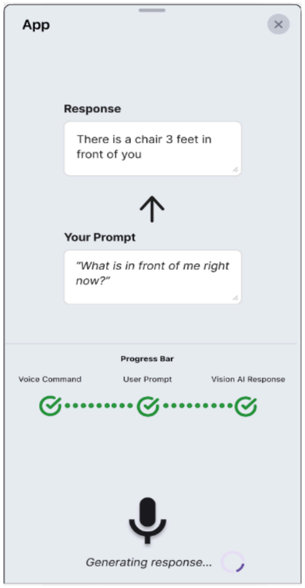
The main challenge with this approach is that a Wi-Fi network usually isn’t the most reliable for portable devices. Fortunately, the Oak-D IoT has the capability to run as an “access point”, meaning it can create a portable network it to connect to other devices (Luxonis, 2025).

In terms of the actual smartphone application, it contains a few key functions, although acting as a messenger rather than performing many complex tasks. Besides collecting data from the camera, the first function is to record a user’s requests and to convert the audio to text. The second function is to connect to a cloud-based GPT model, sending both the visual data and the user’s request to the GPT-model. The third and final function is to collect the response from the GPT and convert it to audio via text-to-speech (TTS).

With that being said, there are a lot of different moving parts here, which can bring into question the difficulty of setting up an application like this. Fortunately, references from other designs and prototypes can demonstrate this as more than just a hypothetical idea. In this case, Vuzix is a very helpful resource for outlining and developing a fully functional smartphone companion app. For background, they have developed their own camera glasses (M400/M4000/Blade) devices, which is a framework that other developers such as Envision have based their architecture on. It shares almost every feature with the design being proposed in this paper (object detection, voice commands, cloud-based GPT), with the only exception being depth estimation.

On their website, Vuzix provides clear guidelines for creating a custom development project, containing extensive technical details such as APIs for the connectivity pipeline of the device. Most notably, it provides instructions for working with their custom SDKs for implementing more complex tasks like speech recognition and barcode scanning, even providing code examples. However, since these SDKs can only be run on Android devices (as they themselves are Android-based), it's best to use these instructions more so as a reference design, if a cross-platform functionality is the goal here.

Considering this is a mobile application, the ideal scenario would be to use React Native, as it would allow this desired functionality across both iOS and Android. Using React Native specifically would also come with a large selection of APIs that can natively run on the application and would be able to support important functions such as mic recordings, STT/TTS, and networking with other devices.



There is a chair 3 feet in front of you

"What is in front of me right now?"

Generating response...

Figure 4, Potential User Interface design for application

A final thing to consider with the application is making it as accessible and easy to use as possible for its visually impaired users. Even if the device runs perfectly, poor UX/UI design can effectively ruin the success of the app, especially since blind people will be the main group to use this. Luckily, there are guidelines in place that can help steer development of the UI in the right direction. The WCAG (Web Content Accessibility Guidelines), separated into 3 levels (A/AA/AAA), measures how usable an app is for those with impairments or disabilities. Again, since blind people are the main demographic, it's crucial to aim to achieve at least Level AA in this case.

To achieve Level AA, an app must have certain core features such as screen reader support, text scaling, high contrast/dark mode for the interface, and clear audio instructions/feedback. Other features that are helpful to have include haptic feedback, voice-controlled input (which is already included), and customizable UI (Sakharchuk, 2025). In a scenario where this gets officially released on the market, it's important to be mindful of these users' needs and to possibly further research ways to make the application more convenient for them.

5.4: Cloud-based GPT

Since the only two things being sent to this GPT are simply object position (in. json) and a text prompt there isn't much demand on the actual algorithm to interpret the user's surroundings. For this reason, a simple ChatGPT extension would suffice.

Much like YOLO, OpenAI has many different models for ChatGPT that can be accommodated for various functions. In this case, the GPT-4o mini is best suited for a real-time API, only costing $0.60 for 1 million input tokens. According to OpenAI's own website, a transcript of the U.S. Declaration of Independence only costs around 1,800 tokens, so it's able to handle lots of smaller requests before possibly running out (OpenAI, 2025). Fortunately, this very well could be cheap enough that the users wouldn't need to pay to use the app/API (and instead could be covered by the developers/company), given they aren't using it to an extreme degree.

Discussion

Through researching into each individual component, such as their performance capabilities and compatibilities with other software/hardware, this concept for a device is backed up by various other sources that have either studied this idea or have created a similar device.

While this idea isn't completely original or groundbreaking, running the device almost completely on the cloud allows increased portability and a cheaper cost to produce. One consideration to be made is the reliance on Wi-Fi and cellular data, which aren't completely stable even with an established connection. Another consideration is the amount of different processes running at once, whether it's the DepthAI/YOLO running on the camera, or the LLM on the cloud. This can result in a larger performance demand on the overall system and lead to a large power consumption or even bottlenecking in a certain area. It's also harder to troubleshoot and isolate the issue when there's more complex components in a system such as this one.

With the software application, there are already established precedents and methods in place in order to develop it on a smartphone with the necessary features. However, it's equally important to consider UX/UI design concepts, especially when concerning those with visual impairments, who are likely to be the main demographic for this product.

Conclusion & Recommendations

The current plan for this prototype is as follows:

1. Stereo camera – Oak-D IoT ($250)
2. Object detection AI – YOLO_V8
3. Smartphone application
4. Cloud-based ChatGPT model

The results of this point towards a plausible prototype with all 4 components being able to link together to create a single device. Another conclusion that can be drawn is the price point, as it only comes out to roughly ~$250 total, while competitors can have prices up to 10 times that. It's a testament to the cost efficiency of products that run on the cloud, though it does come with some important considerations.

The smartphone application itself will most likely run on React Native, utilizing built-in API support to achieve all of the necessary functions of the application. It will also need to heavily utilize certain UX/UI design features in order to achieve a high degree of accessibility for the target demographic.

The biggest limitation of this study is the actual performance of the device, on both the hardware and software side. A lot of data has been provided to give a general idea, but it isn't concretely known until the device is created. Once the product is finished, future research could be conducted on ways to optimize the product, create new features to differentiate itself, and the improve the general user experience.

References

- Waisberg, E., Ong, J., Masalkhi, M., Zaman, N., Sarker, P., Lee, A. G., & Tavakkoli, A. (2024). Meta smart glasses-large language models and the future for assistive glasses for individuals with vision impairments. *Eye (London, England)*, *38*(6), 1036–1038. https://doi.org/10.1038/s41433-023-02842-z
- G. Gollagi, S., D. Bamane, K., Manish Patil, D., B. Ankali, S., & M. Akiwate, B. (2023). An innovative smart glass for blind people using artificial intelligence. *Indonesian Journal of Electrical Engineering and Computer Science*, *31*(1), 433. https://doi.org/10.11591/ijeecs.v31.i1.pp433-439
- Roboflow. (2023, January 10). *Yolov8 object detection model: What is, how to use*. Object Detection Model: What is, How to Use. https://roboflow.com/model/yolov8
- The4. (n.d.). *Oak-D lite*. Luxonis. https://shop.luxonis.com/products/oak-d-lite-1?variant=42583102456031

- Kumaran, J. (2025, February 27). *Object detection on edge device - oak-D*. LearnOpenCV. https://learnopencv.com/object-detection-on-edge-device/
- Apple. (n.d.). *Transcribing speech to text*. Transcribing Speech to Text. https://developer.apple.com/tutorials/app-dev-training/transcribing-speech-to-text
- *API pricing*. OpenAI. (n.d.). https://openai.com/api/pricing/
- *What are tokens and how to count them? | openai help center*. OpenAI. (n.d.-b). https://help.openai.com/en/articles/4936856-what-are-tokens-and-how-to-count-them
- Smith, Z. (2023, December 12). *Overview*. Developer Resources. https://support.vuzix.com/docs/overview-22
- Sakharchuk, S. (2025, May 29). *A guide to building apps for Blind People in 2025*. Designing Apps for Visually Impaired in 2025: Inspiring Examples and Practical Tips. https://interexy.com/building-app-for-blind-people-a-step-by-step-guide-to-creating-accessible-and-user-friendly-navigation-app/