



An Integrated Mobile Vision System for Enhancing the Interaction of Blind and Low Vision Users with Their Surroundings

Jin Chen¹ ^a, Satish Ramnath², Tyron Samaroo², Fani Maksakuli¹, Arber Ruci¹,
E'edresha Sturdivant¹ and Zhigang Zhu^{2,3} ^b

¹Nearabl Inc., New York, NY 10023, U.S.A.

²Computer Science Department, The City College of New York - CUNY, New York, NY 10031, U.S.A.

³PhD Program in Computer Science, The Graduate Center - CUNY, New York, NY 10016, U.S.A.

Keywords: Mobile Application, Voice Interaction, 3D Finger Recognition, 3D Object Detection, Assistive Computer Vision.

Abstract: This paper presents a mobile-based solution that integrates 3D vision and voice interaction to assist people who are blind or have low vision to explore and interact with their surroundings. The key components of the system are the two 3D vision modules: the 3D object detection module integrates a deep-learning based 2D object detector with ARKit-based point cloud generation, and an interest direction recognition module integrates hand/finger recognition and ARKit-based 3D direction estimation. The integrated system consists of a voice interface, a task scheduler, and an instruction generator. The voice interface contains a customized user request mapping module that maps the user's input voice into one of the four primary system operation modes (exploration, search, navigation, and settings adjustment). The task scheduler coordinates with two web services that host the two vision modules to allocate resources for computation based on the user request and network connectivity strength. Finally, the instruction generator computes the corresponding instructions based on the user request and results from the two vision modules. The system is capable of running in real time on mobile devices. We have shown preliminary experimental results on the performance of the voice to user request mapping module and the two vision modules.

1 INTRODUCTION


Traveling and interacting with objects or places are common activities in people's everyday life. Vision plays an essential role in gathering the necessary information for people to perform these activities, such as determining walkable paths and locations of objects or places of interest. However, people who are blind or have low vision (BLV) have limited visual information, thus increasing their challenges while undertaking these activities, especially in unfamiliar environments.


Numerous assistive technologies have been developed to help bridge this visual gap for BLV users. However, most of these applications focus on providing virtual assistance in navigation and obstacle avoidance for BLV users. They face further problems

during traveling, especially when interacting with objects, such as opening the door, pressing the correct elevator button, or grasping target objects. These tasks are trivial for normal-sighted people but require a lot more effort from BLV individuals.

Delivering the object locations to BLV users is also challenging, as it is difficult to convey the spatial location solely through voice instructions. For instance, instructing a BLV user that the target object is located 12 degrees to the right of the camera's direction and 5 meters away, most BLV users would not know how much they need to move for the 12 degrees. Furthermore, the individual's facing direction may not align with that of the camera's, exacerbating the difficulty in conveying spatial information.

Traditional assistive solutions for BLV users often require bulky and expensive wearable devices, which can limit usability and accessibility. This work proposes a novel low-cost integrated vision system, which aims to overcome these limitations,

^a  <https://orcid.org/0000-0003-1810-3828>

^b  <https://orcid.org/0000-0002-9990-1137>

by utilizing mobile devices that users already have, such as iPhones, to reduce costs and increase user-friendliness.

The system is composed of several key components: a voice interface to determine the user's request through their voice input and provide voice feedback, an instruction generator to create feedback based on the user request and detected results from the two 3D vision modules, and a task scheduler for managing the computational resources of the vision modules. These components are all executed on a user's mobile device, enabling the whole system to work in near real-time. The 3D vision modules are comprised of a 3D object detection module for localizing the target objects, and an interest direction recognition module for determining the user's interest direction. The interest direction recognition module is particularly unique in its ability to determine the user's region of interest based on either the device's camera orientation or the user's finger-pointing direction in 3D, including partial hand capture. These modules expand the range of possible interactions for BLV users, allowing for more accurate and versatile navigation and exploration. Furthermore, these two 3D vision modules can be executed either on the user's mobile device or in the cloud, managed by the task scheduler based on available computational resources and network connectivity strength.

This work presents five main contributions to vision engineering:

- An interest direction recognition module that determines the user's region of interest using either the device's camera orientation or the user's finger-pointing direction estimated in 3D, even with partial hand captured.
- A task scheduler that coordinates the computational resources of the two local modules and two web services for the two 3D vision modules, reducing the computational burden on the mobile device.
- An instruction generator that generates feedback based on the user's request and detected results from the two 3D vision modules, handling the potential collision of the feedback.
- A voice interface allows for seamless interaction with the user, using a customized user request mapping module to interpret the voice input.
- A mobile device-based integrated system, that enables BLV individuals' independence and quality of life by enabling them to better explore and interact with their surroundings, without additional specialized devices.

The paper is organized as follows. Section 2 discusses the state-of-the-art visual assistive tools for BLV users. Section 3 provides an overview of the proposed system. Section 4 presents the details of the two 3D vision modules used in the system. Section 5 describes the implementation of the three major components of the system. Section 6 showcases the experimental results of the system. Finally, Section 7 presents the conclusion and future work directions.

2 RELATED WORK

Over the past decades, there has been a significant development of assistive technologies aimed at enhancing the safety and independence of BLV individuals in performing daily tasks. These technologies can be broadly categorized into two types: mobility and reading assistance.

Smart canes, wearable devices, and mobile applications are common products in mobility assistance to help BLVs get around the space and avoid obstacles. Smart canes usually consist of multiple sensors to monitor the environments, such as the ultrasonic sensors that are used in (Ghani and Zariman, 2019) and (Guerrero et al., 2018) to detect the distance of the obstacles to canes. Some research integrates computer vision systems and ultrasonic sensors by attaching the camera to the cane to detect different objects, such as stairs and people through object recognition and facial recognition (Majeed and Baadel, 2016; Bouhamed et al., 2013). N. Ahmad et al. (Ahmad et al., 2018) integrate multiple sensors including ultrasonic sensors, accelerometer, vibration motor, infrared sensor, and microcontrollers to minimize false detections by filtering through dynamic feedback compensators and optimizing the sensors' locations. Hearsee Mobility¹ developed smart canes that communicate with a mobile application which provides navigational instructions and indicates the nearby points of interest to BLV users.

Besides using smart canes, there are also vision-based navigation solutions for BLV users. Visual simultaneous localization and mapping (Bai et al., 2018) was used to detect objects from a single camera with various techniques, to develop a dynamic sub-goal selecting-based route following algorithm to help users avoid obstacles along the navigation. (Hakim and Fadhil, 2019) proposed a wearable glass prototype with an RGB-D camera and ultrasonic sensors to integrate the detection inputs while detecting small and transparent obstacles. Similarly, P. Rajendram

¹<https://www.hearseemobility.org/>

et al. also proposed using a novel approach with smart glasses for the vision process and audio feedback to avoid surrounding obstacles (Rajendran et al., 2020). Although these applications can help BLVs get around safely, they do not provide enough information to assist them understand their surroundings better or in performing any interaction they need to reach their destination.

Reading assistance tools for BLV include OrCam², which produces wearable technology solutions that are capable to attach to user's glasses to capture their intent view. It processes the captured view to help users with things like reading text in the scene, recognizing faces, identifying objects, money barcodes, and more. Microsoft developed Seeing AI³, which provides a similar solution as OrCam, but instead of requiring an additional wearable device, it is delivered through the mobile application which lower the costs for the users (Granquist et al., 2021). Besides the pure text reading assistance, there are also applications to help BLV users to understand printed images. iReader (Jothi et al., 2022) combines the convolution neural network with the long short-term memory network to generate the image description.

Our approach stands out from existing approaches by enabling BLV users to use both their voice and hand to enhance their understanding of their surroundings and facilitate interaction with their environment. In line with Microsoft's solution, our method is a mobile application that does not necessitate the acquisition of a separate wearable device.

3 SYSTEM OVERVIEW

The proposed system (Fig. 1) works on an iOS application that assists BLV users to interact with their surrounding objects through their voice input and/or hand gestures. The system supports four primary system operation modes. First is the **exploration mode**, which informs users about objects in their interest direction based on their finger-pointing direction or camera orientation. Second is the **search mode**, which obtains the target object according to the user's voice request and searches for the target object in the scene. The third is the **navigation mode**, which guides users to reach the target object based on their interest direction. Last is the **settings adjustment mode**, which allows users to customize the system's settings, such as voice feedback volume and speed.

²<https://www.orcam.com/>

³<https://www.microsoft.com/en-us/ai/seeing-ai>

The system contains two 3D vision modules (3D object detection and interest direction recognition) and three major components (voice interface, task scheduler, and instruction generator). **3D object detection** module detects the 3D position and size of the objects within the captured frame and will rely on our previous work (Chen and Zhu, 2022). **Interest direction recognition** module determines the user's interest direction by analyzing either the camera's orientation or the detected user's finger-pointing direction in the capture camera frame.

Voice Interface component interprets the user's voice input and identifying the corresponding system operation mode. Before passing the request to the task scheduler, it confirms the matched user request with the user. The system also provides voice feedback to the user with the instructions generated based on the user's requests and computed results from the two vision modules. **Task Scheduler** component schedules and allocates resources for the computation of the two vision modules (**3D object detection** and **interest direction recognition**) based on the user request. The two vision modules have their own web service and a local module that can run on the user's device. The task scheduler will determine where the computation will take place based on the available device resources and network connectivity strength. Finally **Instruction Generator** component generates instructions based on the user request and results computed from the two vision modules, which include the 3D information of the detected or target object and the user's interest direction. The instructions are then delivered to the user through the voice interface.

4 3D VISION MODULES

Understanding the 3D location of surrounding objects and the user's interest direction in the scene is essential for assisting BLV users' interaction with the surrounding space.

4.1 3D Object Detection

The 3D object detection module detects the 3D locations, sizes of the objects and people that exist in the user's camera view. This module is built based on our previous work (Chen and Zhu, 2022) that utilizes deep-learning based 2D object detector to first obtain the 2D bounding boxes of the objects in the camera frame, then uses the ARKit point cloud or depth map, gathered by the LiDAR sensors on compatible iOS devices, to estimate the rough 3D bounding boxes of the detected objects based on the object's 2D bound-

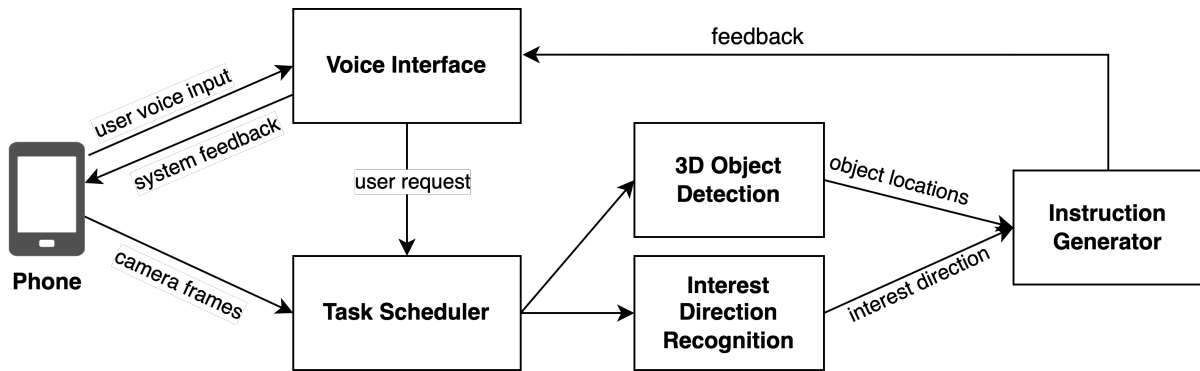


Figure 1: Overview of the system architecture for improving BLV interaction with surrounding objects, featuring a voice interface, a task scheduler for coordinating 3D object detection and interest direction recognition modules, and an instruction generator for customized feedback based on user request.

ing boxes. To refine the object’s 3D bounding boxes across consecutive frames, the module then applies a non-maximum suppression (NMS) technique.

4.1.1 Interest Object Filtering

Based on the target object requested by the user or the user’s interest direction, a filter is adapted to track only the interest objects across the frame and/or provide correspondent object information. To filter objects based on the user’s interest direction in 3D, the system first eliminates objects with centroids behind the user’s initial position (P_i , details described below in Section 5.3) with respect to the user’s interest direction at the XZ-plane. Second, the system computes the y-direction difference ($\Delta\theta$) between each of the remaining objects and the user’s interest direction, and filters out the objects whose $\Delta\theta$ values are less than the angular threshold (α). Next, the system ranks the remaining objects based on each object’s delta y-direction ($\Delta\theta$) the shortest distance (ΔD) to the user’s interest direction, and the object’s 3D size (O_s), using equation 1, with three weighted parameters (a, b, c).

$$\text{Rank}(O) = a * \Delta\theta_o + b * \Delta D_o + \frac{c}{O_s} \quad (1)$$

The object with the lowest ranking will be the closest object with respect to the user’s interest direction.

4.2 Interest Direction Recognition

Existing approaches in providing spatial information to BLV users through voice feedback can be insufficient, particularly when guiding them to move in a specific direction. Although many existing systems utilize the camera orientation to indicate the user’s interest direction, this may not always correspond to the

user’s actual direction of interest due to the misalignment of the camera’s facing direction and the user’s gaze direction. To address this issue, the system includes a 3D finger-pointing direction detection module that uses hand gestures captured by the camera frame to determine the user’s interest direction given by their finger-pointing direction. Here we only assume that the hand gestures could be partially captured.

The system will passively use camera orientation as the interest direction indicator and switch to finger-pointing direction detection when requested by the user or when the object detection module detects the user’s hand in the camera frame.

4.2.1 Finger-Pointing Direction Detection

The 3D finger-pointing direction detection module utilizes Swift’s Vision framework⁴ or Mediapipe hand landmark model (Zhang et al., 2020) to detect the 21 key landmarks of the user’s hand from the input camera frame. The key landmarks consist of four joints from each finger and the base of the wrist. The system uses the index finger as the interest direction indicator. The camera view cannot always capture the user’s hand and the system needs to detect at least two landmarks of the index finger to determine the user’s interest direction. In case only one landmark is detected, the system will alert and guide the users to either move their hand in a direction to be captured by the camera or ask the user if they want to change the camera orientation as the interest direction indicator.

⁴<https://developer.apple.com/documentation/vision/vndetecthumanhandposerequest>

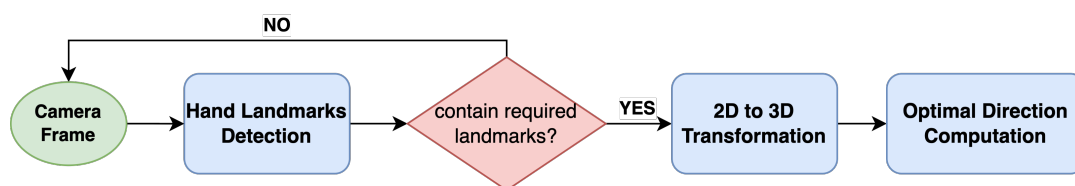


Figure 2: The process of determine the user's finger-pointing direction from captured camera frame.

4.2.2 Projecting Finger-Pointing Directions from 2D to 3D

Once the index finger landmarks are detected in 2D coordinates, the system will project that into 3D camera coordinates through ARKit⁵, which is the same framework used in our object detection module (Chen and Zhu, 2022). The 3D coordinates of each landmark will be obtained either through estimation with AR point cloud (a set of the notable features points collected by ARKit) of the corresponding camera frame or using depth value collected by LiDAR sensor available in selected iOS devices. If there are three or more landmarks detected, the system will compute the 3D orthogonal distance regression (ODR) line using singular value decomposition to determine the optimal interest direction that the user's index finger is pointing toward. However, if there are only two landmarks detected, the optimal interest direction would be the direct connection between the two landmarks.

5 SYSTEM IMPLEMENTATION

5.1 Voice Interface

The voice interface plays a crucial role in enabling BLV users to utilize the system functionalities with minimal need for manual interaction with the application. The voice interface comprises two stages: the first stage involves receiving the user's voice input and matching it with the system operation modes, while the second stage involves delivering the system feedback to the users.

The system uses the Whisper speech recognition model (Grozdić et al., 2017) to convert the user's voice input into text, which is then processed to match the user's request with a system support mode. The first step involves removing stop words and punctuation from the resulting text, as these typically do not contain useful information and removing them can reduce the amount of irrelevant or extraneous information that must be processed. Next, a part-of-speech (POS) tagging algorithm (Jurafsky and Martin, 2008)

⁵<https://developer.apple.com/augmented-reality/arkit/>

is applied to the cleaned text. The customized user request mapping module then separates the text into two main components: the *mode* the user wants to perform and the *task* to be accomplished. A random forest model is used to classify the text component of the *mode* into one of four primary system operation modes (i.e., exploration, search, navigation, and settings adjustment). The system then confirms the determined user's request with the user before forwarding it to the task scheduler.

To deliver the system feedback to the user, the system uses the Swift speech synthesis module⁶, where the speed of speech and volume can be customized by the user. Each system feedback contains two values, priority and type, for handling the collision. The collision in our system represents the time when the system is still in the process of delivering the current feedback and there is another incoming feedback that needs to be delivered. In this case, if the incoming feedback's priority is higher than the current feedback, then the system will stop giving the current feedback and deliver the incoming feedback. If both feedbacks have the same priority value but different types, it will also replace the current feedback. However, if the feedbacks are the same type, then the current voice feedback will be updated by the incoming feedback, which usually regarding the object's location information. If incoming feedback's priority is lower than the current feedback, it will be ignored.

5.2 Task Scheduler

Running 3D object detection and interest direction recognition modules on a single mobile device could exceed the computation capability that the device can support, which can negatively affect real-time performance and cause battery drainage. Therefore, two web services were created to host the two vision modules separately. Although both modules are still embedded in the application for situations where a user has poor network connectivity, hosting the modules on separate servers helps to reduce the computational burden on the device.

⁶https://developer.apple.com/documentation/avfoundation/speech_synthesis

The task scheduler determines the required module(s) and where the computation will take place. Typically, only one module is required at a time. For instance, the object search mode requires only 3D object detection, while navigating to a pre-detected static object requires only interest direction recognition. However, some situations require simultaneous execution of two modules, such as, navigating the user to target objects requires both the target object's 3D position and the user's interest direction. After determining the module(s) required based on the user request, the task scheduler will allocate computational resources based on the strength of the network connectivity.

Using web services requires stable and strong network connectivity to transmit data to the cloud, while the processing time of each module can easily be adjusted by regulating the processing power in the web services. The system continuously evaluates the network connectivity strength and categorizes it into four levels: poor ($\leq 1\text{MB/s}$), fair ($\leq 10\text{MB/s}$), good ($\leq 20\text{MB/s}$), and excellent ($> 20\text{MB/s}$). Based on the connectivity strength level, the task scheduler will determine whether to run the modules concurrently in the web services, one in the cloud and one on the device, or both on the device.

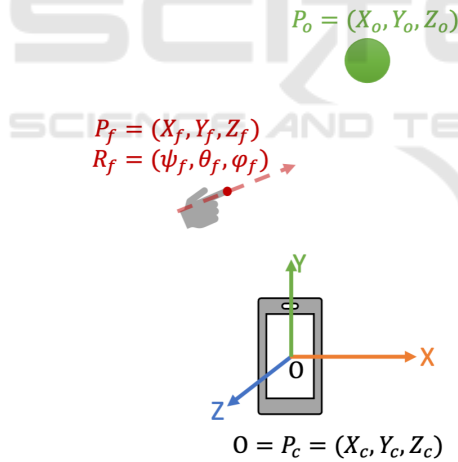


Figure 3: Position of the camera, finger landmarks and object in camera coordinate system.

5.3 Instruction Generator

The system feedback for the user would either instruct the user to move toward the target object or convey the location of the target object with respect to the user's initial position (P_i) and interest direction (R_i).

The detected object's 3D position (P_o) by the 3D object detection module is with respect to the camera coordinate system. Based on the method used for the user's interest direction recognition, the ini-

tial position can be either the camera position (P_c) or the detected index fingertip's 3D position (P_f), and the interest direction can be either camera orientation (R_c) or the computed optimal finger-pointing direction (R_f). Four pieces of information can be computed based on different situations, which are 1) the direction in the y-axis (left or right) using equation 2, 2) the distance between the objects and initial position in XZ-plane using equation 3, 3) direction in X-axis (up or down) using equation 4, 4) the distance between the objects and initial position in Y-axis using equation 5.

$$direction(Y) = \theta_i - \text{atan2}(Z_o, X_o) \quad (2)$$

$$distance(X, Z) = \sqrt{(X_i - X_o)^2 - (Z_i - Z_o)^2} \quad (3)$$

$$direction(X) = \psi_i - \text{atan2}(Z_o, Y_o) \quad (4)$$

$$distance(Y) = X_o - X_i \quad (5)$$

If the user request is obtaining the object location, and if the distance between the object (P_o) and the user's initial position (P_i) in the XZ-plane is above a threshold (i.e., 50cm), the system feedback will be the distance in the XZ-plane along the direction on the Y-axis, otherwise it will be the distance on the Y-axis along the direction of the X-axis. If the user request is to obtain the direction to the object, only one piece of information will be given at a time. The system will validate the four pieces of information in order along their corresponding threshold. If it is less than the threshold, the corresponding information will be provided to the user.

6 EXPERIMENTAL RESULTS

6.1 Voice to User Request Mapping Evaluation

Classifying the user's voice input into one of the four primary system operation modes is a challenging task due to the variability in how users express their requests. The customized user request mapping module currently classifies the user's voice input into four categories, exploration, search, navigation, and settings adjustment. Additionally, there are multiple Whisper speech recognition models, each with different computation speeds and accuracy in converting the audio to text. To evaluate the performance of our mapping module, we tested 40 audio samples with lengths

ranging from 1 second to 4 seconds using the base Whisper speech recognition model, which has a size of 139 MB.

Table 1: Voice to user request mapping module performance with Whisper base model.

User Request	Precision	Recall	F1-Score
Exploration	0.75	0.75	0.67
Search	0.90	1.00	0.94
Navigation	0.67	0.67	0.65
Settings Adjustment	0.83	0.58	0.67

The test accuracy for the user request mapping is 75.62%, with navigation performing the worst and often being confused with exploration. It is challenging to distinguish between the four types of user requests as users express themselves differently and there is a lack of training samples. However, there is a confirmation step with the user to validate the predicted user request. Moreover, with the user's consent, the system can collect the user's voice input and corresponding requests over time to refine the model's performance.

6.2 3D Vision Module Performance

An iOS application was created to evaluate the performance of the proposed system, especially the task scheduler for managing the computations of the two vision modules. All the experiments are performed in iPhone 13 Pro Max with an Apple A15 Bionic processor that has a 16-core neural engine, 6-core CPU, and 5-core GPU. Both object detection and finger direction recognition web services are hosted on the MacBook Pro with a 2.7GHz quad-core Intel Core i7.

Table 2 displays the time spent on the two vision modules in milliseconds (ms). In the experiment, the YOLOv5 medium model (Glenn et al., 2022), which has a model size of 85.1MB, was utilized for object detection in both the iOS app and web services. For finger direction recognition, the Swift's Vision framework was used in the iOS app, while the Mediapipe hand landmark model was utilized in the web services. The image resolution for both vision modules was set to 640x480.

The mobile run time is much less than the web services run time, which is due to the consumption of processing power in the mobile device and additional data transmission time to the cloud. The data ($\leq 1\text{MB}$) sent to the web services includes the camera frame, AR point cloud or depth map, camera intrinsic and extrinsic parameters. There is a time difference

in the data transmission time for both modules when sending a similar amount of data to the web services through the same network. This is due to the experiments performed during different times where there is a difference in the network upload speed ($5 \sim 10$ MB/s). On average, the system can support real-time performance of updating the feedback message in 10 frames per second (fps) with all modules running on the mobile device and 2 fps running through the web services. However, in real-world scenarios, the web services can be hosted in a more resource-rich environment (e.g., with GPU instances) to reduce the computational time.

In addition, running the modules in the iOS device consumes more battery. With two modules running simultaneously in the iOS device, it affects the ARKit performance over time, leading to inaccurate data collected for 3D projection. Therefore, it is essential to have the task scheduler allocate resources for the computation to coordinate with the network single strength and application performance in the iOS device to achieve a feedback message update speed of 5 fps or more.

7 CONCLUSION

In this paper, we proposed a mobile device-based integrated system to enhance the interaction of BLV users with their surroundings. The system includes a 3D finger direction recognition module that allows users to use hand gestures as interest direction indicator beside the camera orientation, a task scheduler for flexible computational allocation that minimizes the computational requirements needed in the mobile device and ultimately allow the system to have real-time performance, and a voice interface with a customized user request mapping module allows users to interact with the app using their voice. The system can potentially collect more samples of the user's voice input with the user's consent, to improve the user request mapping module performance.

A potential problem could arise if the camera frame captures multiple hands or another person's hand. To overcome this issue, the system could incorporate a distance filter and a tracking module to filter out false detections.

In the future, we would like to further develop models to guide users to perform actions for their tasks, such as how they should interact with the objects and validate their action. Additionally, we want to integrate this system with our previous indoor navigation system to increase the BLV user's accessibility in indoor exploration and navigation. This

Table 2: Comparison of run time for the two 3D vision modules (object detection and finger direction recognition in the mobile device and web services.

Module	Mobile Run Time (ms)	Web Services Run Time(ms)		
		Data Transmission	Cloud Computation	Total
Object Detection	73.01±4.45	267.05±98.19	201.36±26.65	468.41±104.40
Finger Direction Recognition	56.84±4.51	344.11±126.45	48.69±10.37	392.81±130.17

would require us to expand on the existing user request mapping module and task scheduler to incorporate the necessary changes for future integration.

ACKNOWLEDGEMENTS

The work is supported by the US National Science Foundation (#2131186, #2118006, #1827505, #1737533, and #2048498), ODNI Intelligence Community Center for Academic Excellence (IC CAE) at Rutgers (#HHM402-19-1-0003 and #HHM402-18-1-0007) and the US Air Force Office for Scientific Research (#FA9550-21-1-0082).

REFERENCES

- Ahmad, N. S., Boon, N. L., and Goh, P. (2018). Multi-sensor obstacle detection system via model-based state-feedback control in smart cane design for the visually challenged. *IEEE Access*, 6:64182–64192.
- Bai, J., Lian, S., Liu, Z., Wang, K., and Liu, D. (2018). Virtual-blind-road following-based wearable navigation device for blind people. *IEEE Transactions on Consumer Electronics*, 64(1):136–143.
- Bouhamed, S. A., Kallel, I. K., and Masmoudi, D. S. (2013). New electronic white cane for stair case detection and recognition using ultrasonic sensor. *International Journal of Advanced Computer Science and Applications*, 4(6).
- Chen, J. and Zhu, Z. (2022). Real-time 3d object detection and recognition using a smartphone [real-time 3d object detection and recognition using a smartphone]. In *Proceedings of the 2nd International Conference on Image Processing and Vision Engineering-IMPROVE*.
- Ghani, F. A. and Zariman, A. (2019). Smart cane based on iot. *International Journal of Education, Science, Technology, and Engineering*, 2(1):12–18.
- Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, TaoXie, Jiacong Fang, imyhxy, Kalen Michael; Lorna, Abhiram V, Diego Montes, Jebastin Nadar, Laughing, tkianai, yxNONG, Piotr Skalski, Zhiqiang Wang, Adam Hogan, Cristi Fati, Lorenzo Mammana, AlexWang1900, Deep Patel, Ding Yiwei, Felix You, Jan Hajek, Laurentiu Diaconu, Mai Thanh Minh (2022). *ultralytics/yolov5: v6.1 - TensorRT, TensorFlow Edge TPU and OpenVINO Export and Inference*. [Online]. Available from: <https://doi.org/10.5281/zenodo.3908559/>.
- Granquist, C., Sun, S. Y., Montezuma, S. R., Tran, T. M., Gage, R., and Legge, G. E. (2021). Evaluation and comparison of artificial intelligence vision aids: OrCam myeye 1 and seeing ai. *Journal of Visual Impairment & Blindness*, 115(4):277–285.
- Grozdić, Đ. T., Jovičić, S. T., and Subotić, M. (2017). Whispered speech recognition using deep denoising auto-encoder. *Engineering Applications of Artificial Intelligence*, 59:15–22.
- Guerrero, J. C., Quezada-V, C., and Chacon-Troya, D. (2018). Design and implementation of an intelligent cane, with proximity sensors, gps localization and gsm feedback. In *2018 IEEE Canadian Conference on Electrical & Computer Engineering (CCECE)*, pages 1–4. IEEE.
- Hakim, H. and Fadhil, A. (2019). Navigation system for visually impaired people based on rgb-d camera and ultrasonic sensor. In *Proceedings of the International Conference on Information and Communication Technology*, pages 172–177.
- Jothi, G., Azar, A. T., Qureshi, B., and Kamal, N. A. (2022). ireader: An intelligent reader system for the visually impaired. In *2022 7th International Conference on Data Science and Machine Learning Applications (CDMA)*, pages 188–193. IEEE.
- Jurafsky, D. and Martin, J. H. (2008). Speech and language processing: An introduction to speech recognition, computational linguistics and natural language processing. *Upper Saddle River, NJ: Prentice Hall*.
- Majeed, A. and Baadel, S. (2016). Facial recognition cane for the visually impaired. In *International Conference on Global Security, Safety, and Sustainability*, pages 394–405. Springer.
- Rajendran, P. S., Krishnan, P., and Aravindhar, D. J. (2020). Design and implementation of voice assisted smart glasses for visually impaired people using google vision api. In *2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)*, pages 1221–1224. IEEE.
- Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C.-L., and Grundmann, M. (2020). Mediapipe hands: On-device real-time hand tracking. *arXiv preprint arXiv:2006.10214*.