ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

Echo Vision: AI Voice Assisted Navigation for Visually Impaired

Bala Velan M[1]; Jayanth Balraj A[2]; Jerusha Angel K[3]; Oswalt Manoj[4]

[1,2,3] UG Scholars, Department of Computer Science and Engineering, Sri Krishna College of Engineering and Technology

[4]Associate Professor, Department of Computer Science and Engineering, Sri Krishna College of Engineering and Technology

Publication Date: 2025/09/23

Abstract: Echo Vision is a mobile assistive navigation system that uses augmented reality (AR) and edge computing to guide visually impaired users. We use ARCore for real-time camera pose tracking on smartphones that support it. If a device does not support ARCore, we use ORB-SLAM3. The system constantly checks if the environment is indoor or outdoor using a deep scene recognition model (Places365). This information helps select the correct object detection method. A YOLOv8 model is used for object detection indoors, and an SSD Mobile Net V3 detector is used outdoors for important obstacles. The phone sends video frames to a local server over Wi-Fi, and the server returns identified objects with their locations and distance ranges (for example, 0–1 m or 1–1.5 m). A sliding memory module on the server combines recent detections and creates periodic voice prompts that summarize nearby obstacles. This avoids too many repeated alerts. The client application shows visual overlays of the environment and obstacles and gives voice warnings through Android TextToSpeech, with vibration alerts for objects closer than 1 m. Tests in hallways and outdoor paths show that EchoVision can adjust its frame rate based on changes in the scene. This helps manage network use. This paper describes the architecture, implementation, and performance of Echo Vision. It also explains the current features and future steps for an edge-cloud assistive platform.

Keywords: *Accurate Localization, Deep Learning, Comprehensive Route Planning, Specialized Detection, Fragmented Coverage, Indoor Navigation, Robust Mapping.*

How to Cite: Bala Velan M; Jayanth Balraj A; Jerusha Angel K; Oswalt Manoj (2025) Echo Vision: AI Voice Assisted Navigation for Visually Impaired. *International Journal of Innovative Science and Research Technology,* 10(9), 1228-1234. https://doi.org/10.38124/ijisrt/25sep880

I. INTRODUCTION

Many people with visual impairments face challenges in navigating. Traditional tools, such as a white cane, help detect obstacles nearby but give limited scene awareness. Recent progress in computer vision and mobile computing makes it possible to offer a richer understanding of surroundings. Earlier work includes wearable vision systems that provide audio or haptic feedback. For example, Bourbakis et al. [1] presented the Tyflos smart assistant, which uses camera input and voice feedback to describe the user's environment. Lee et al. [2] proposed a system with an RGB-D camera to guide a user indoors and outdoors by providing tactile signals. These solutions often need special hardware or have difficulty running in real-time. There is a need for a phone-based system that can process the environment and deliver important information without delays or too much distraction.

Echo Vision uses smartphone sensors and processing together with a local server for extra computing power. The system can tell whether the user is indoors or outdoors by using a Places365 scene classification network. It then activates the right object detection model: YOLOv8 for indoor details or an SSD MobileNetV3 model for outdoor obstacles. This approach focuses on obstacles that appear in each environment without forcing the phone to run multiple detection models at once.

Echo Vision uses an edge offloading architecture: the phone captures and sends video frames to a nearby server for deep learning. The server detects obstacles and estimates how close they are, then sends back a short description of what it sees. To prevent too many alerts, the server keeps track of recent detections in a short memory. It sends summarized voice messages every few seconds, so the user does not hear repeated warnings for the same obstacle. This design follows assistive technology guidelines to limit extra information. The user hears these summaries through the phone's speaker and can also see visual outlines (for users with some vision) and feel vibrations for obstacles that are very close. By combining audio, visual, and vibration feedback, EchoVision delivers information in multiple ways.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

Our main contributions are: (1) a smartphone-based aid that uses ARCore or ORB-SLAM3 to track position while running deep learning to understand the environment, (2) a method that recognizes if the user is indoors or outdoors and picks the right object detection model, (3) an edge computing design that sends heavy vision tasks to a server for real-time processing, and (4) a sliding memory approach that summarizes voice alerts to avoid too many messages. We explain our latest design and show test results in real environments. Section II covers related work, Section III describes the system, Section IV shows test results, and Section V concludes with future plans.

II. RELATED WORKS

Early navigation tools for the visually impaired often depended on special hardware, such as ultrasonic sensors or RGB-D cameras, and provided basic feedback like beeps or vibrations. As computer vision improved, researchers started using object detection and scene classification to give more detailed information. The YOLO family of detectors is popular because it can detect objects quickly and with reasonable accuracy. Erdaw et al. [3] used YOLOv2 for real-time detection of potholes and trash bins. Alsultan et al. [4] tried YOLOv7 for recognizing objects around blind users. Chaudhary et al. [5] built a YOLOv3-based system that also estimated object distances for safer navigation. These studies show that single-shot detectors work well in assistive devices, especially with distance estimation. EchoVision builds on these ideas with YOLOv8, which has an anchor-free structure and several model sizes for different deployment needs [6]. By running YOLOv8 on the server, EchoVision uses these methods without needing a powerful phone.

Scene recognition is also important because it tells the system whether the user is inside or outside. Zhou et al. [7] introduced the Places database with many images across many scene types, allowing deep networks to learn to classify scenes. In assistive systems, knowing the scene type helps decide which objects matter. Some previous work did not consider this context. In EchoVision, we use a Places365 CNN to identify indoor or outdoor settings in real-time. Then we switch detection pipelines. This is somewhat like Lee et al. [2], which used different strategies for indoor and outdoor guidance. However, our approach uses software to pick the right detection model and audio feedback.

Visual SLAM and AR frameworks are often used for localization. ARCore can track a smartphone's position by combining camera and inertial data. This can keep track of the user's movement and place virtual information in the right spot. Zhang et al. [9] showed that ARCore can help visually impaired users by giving stable navigation clues. For older devices, ORB-SLAM3 can provide monocular SLAM for both indoor and outdoor areas [8]. EchoVision includes both ARCore (when available) and ORB-SLAM3 (as a fallback) to keep track of the camera's movement.

Feedback to the user is usually through voice or haptics. Spoken information is easy to understand and can include object names and positions. But if the system speaks too often, it can overwhelm the user. Research recommends filtering alerts and focusing on key details. Some systems also add vibrations for close objects, because vibrations can catch the user's attention quickly. In EchoVision, we deliver voice for overall obstacle information and a vibration alert for very close objects. This approach follows the idea of giving general information in audio and using haptics for urgent warnings. Our sliding memory system reduces repeated announcements by only speaking about obstacles that remain relevant over time.

Offloading tasks to a local server or cloud can improve performance and save phone battery, as many deep learning models are large. Network speed must be considered, but local servers often have enough bandwidth for real-time tasks. Our system sends frames to a server, which runs classification, detection, and summarization, and returns results. This design can scale in the future if a cloud service is used, letting multiple users connect while the phone only handles capturing images and playing back audio.

Echo Vision builds on these ideas: AR-based localization [9], environment classification [7], object detection for assistive tasks [3][4][5][6], and a multi-channel interface. By combining them, along with environment-specific detection and summarized feedback, Echo Vision helps people with visual impairments in different conditions. We will now describe the system's design.

Table 1 Literature Survey

S.No	Title of the Research Paper	Authors	Year	Methodology Used	Limitations
1	An Effective Obstacle Detection System Using Deep Learning Advantages to Aid Blind and Visually Impaired Navigation	A. Ben Atitallah et al.	2024	YOLOv5-based obstacle detection system with pruning and quantization	Requires embedded optimization for real-time use
2	YOLO by Ultralytics (YOLOv8)	G. Jocher et al.	2023	YOLOv8 with improved architecture and speed	Requires high computational power for training
3	SSD: Single Shot MultiBox Detector	W. Liu et al.	2016	SSD for object detection using multi-scale feature maps	Struggles with detecting small objects
4	ORB-SLAM3: An Accurate Open-Source Library	C. Campos et al.	2021	ORB-SLAM3 for simultaneous localization and mapping	Performance degrades in dynamic environments

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

5	Searching for MobileNetV3	A. Howard et al.	2019	MobileNetV3 for lightweight deep learning on mobile devices	Lower accuracy compared to heavier networks
6	SURF: Speeded Up Robust Features	H. Bay, T. Tuytelaars, L. Van Gool	2008	SURF for feature extraction and matching	Less accurate than SIFT in some cases; still not optimized for deep learning
7	Distinctive Image Features from Scale-Invariant Keypoints	D. G. Lowe	2004	SIFT (Scale-Invariant Feature Transform) for keypoint detection	Computationally expensive; not suitable for real-time apps
8	You Only Look Once: Unified, Real-Time Object Detection	J. Redmon, S. Divvala, R. Girshick, A. Farhadi	2016	YOLO (You Only Look Once) single-shot object detection	Less accurate than two-stage detectors like Faster R-CNN
9	A Comprehensive Review of Navigation Systems for Visually Impaired Individuals	M. H. Abidi er al.	2024	A review of various navigation systems for visually impaired	Lack of experimental validation
10	Enhancing Accessible Navigation: A Fusion of Speech, Gesture,Sonification for the Visually Impaired	M. Matei and L. Alboaie	2024	Multimodal approach integrating speech commands, gestures, and sonification	Requires extensive user training; potential latency in real-time gesture recognition

III. PROPOSED SYSTEM

The EchoVision system has a client-side app and a server-side backend. We use an edge-offloading model: the phone captures images and sends them out, and the server does the heavy processing.

➢ *Client (Smartphone App)*

The client is an Android app that uses AR for tracking and has assistive features. On phones that support ARCore, it gets camera poses in real-time by combining camera visuals with inertial sensors. This allows it to place virtual markers in a stable way. If ARCore is not available, the app uses ORB-SLAM3 instead. ORB-SLAM3 runs on the phone and calculates the camera path and a sparse map of the environment. We still send frames to the server for detection. With either ARCore or ORB-SLAM3, the phone has a pose estimate. It checks user movements such as walking or turning.

The app captures and sends video frames at a certain rate. It encodes frames (with some size reduction) to Base64 and sends them to the server through a REST API over Wi-Fi. To avoid slow performance, the app can lower the frame rate if the scene does not change much. If the user is moving fast or new obstacles appear, the frame rate goes up. This balance saves bandwidth and processing time but keeps the system responsive.

When the phone receives the server's detection results, it shows bounding boxes and labels (with approximate distance) on the camera view. It also displays an icon or text that says "Indoor" or "Outdoor," along with the detection model used (YOLO or SSD). The app uses Android TextToSpeech to speak the summary from the server, such as, "Indoor. Chair on your left, near one meter." After some time, if new objects appear or distances change, it might say, "Ahead, table about two meters away." The phone also has a minimum time gap between messages to avoid overlapping speech. If an obstacle is within 1 meter, the phone vibrates for a moment to warn the user of something very close.

➢ *Server*

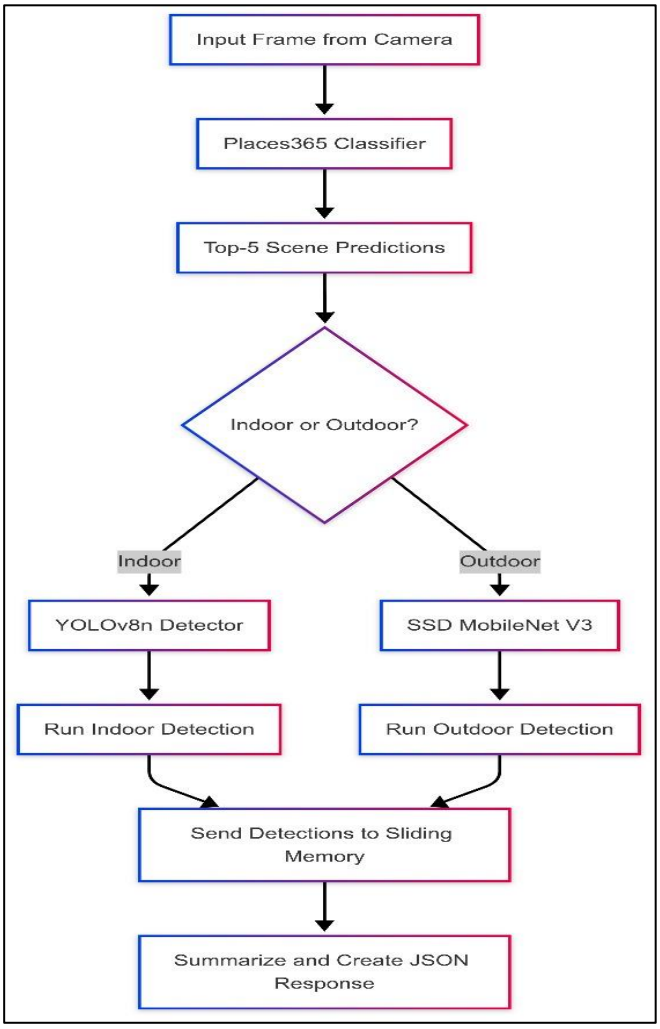

Fig 1 Server Side Processing

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

The server is a Python Flask app on a device in the same network. Its tasks include environment classification, object detection, and summarizing results. When a frame arrives, the server converts the Base64 string into an image. It then uses the Places365 model to decide if the scene is indoor or outdoor. The model outputs probabilities for various scenes. We look at the top predictions and choose indoor or outdoor based on them. If the classification stays consistent over a few frames, the system sets the environment type.

We then pick the right detection model. Indoors, we use YOLOv8. YOLOv8n (trained on COCO) detects many indoor objects. Outdoors, we use SSD MobileNet V3, for real-time tasks and focuses on objects like people and vehicles. Only one model runs at a time, based on the environment type. If the user moves from inside to outside, the system switches models (Fig 1).

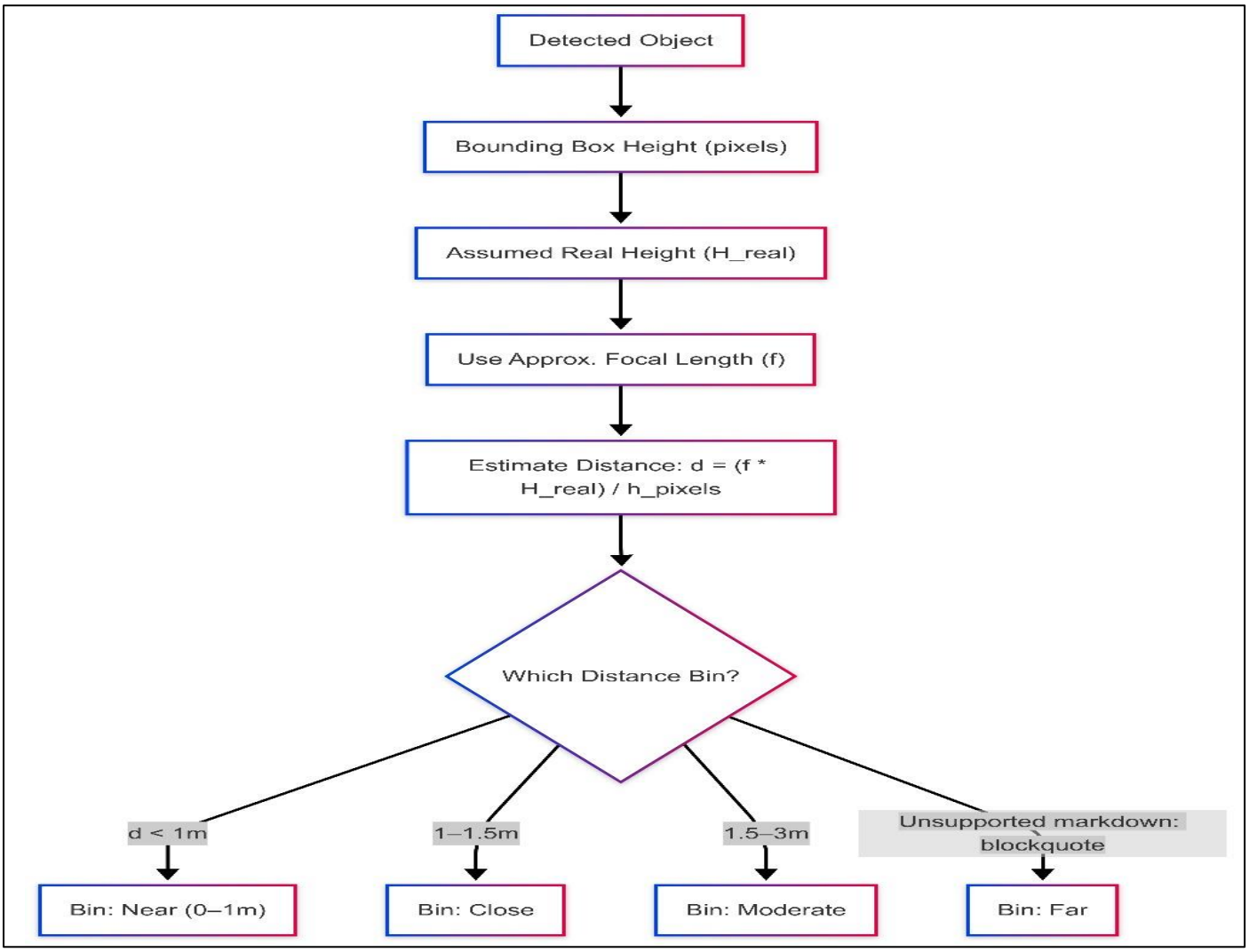

Fig 2 Position Calculation

The detection model produces bounding boxes, labels, and confidence scores. EchoVision estimates each object's distance range by using the object's size on the screen as presented in the following Eq. (1) . We have bins like 0–1 m, 1–1.5 m, 1.5–3 m, and >3 m as seen in *Fig 3*. These bins were chosen through testing. The server makes a list of the objects' labels, box coordinates, and distance bins as seen in *Fig 2*.

- Estimate Distanced = *(f * H_real) / h_pixels* **(1)**
- *f* is the Focal Length
- *H_real* is the Assumed Real Height
- *h_pixels* is the Bounding Box Height

Before sending these results back, the server uses a sliding memory module. It keeps track of objects that appear over a short time and where they are relative to the user. It creates a summary to avoid repeating the same object over and over. If a chair has been detected on the left for several frames, the system mentions it once until something changes.

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

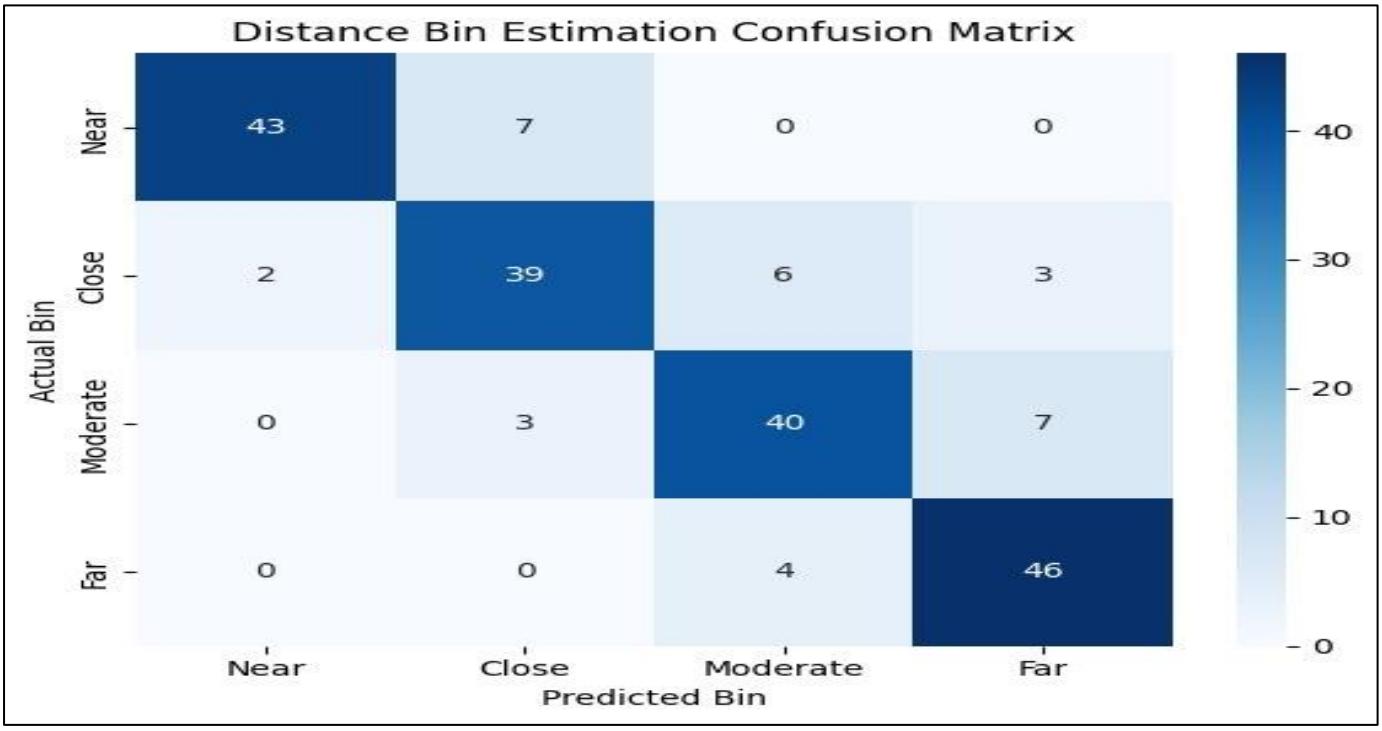

Fig 3 Distance Estimation

The module also highlights objects that are closer or directly in front of the user. It then composes a short message that starts with "indoor" or "outdoor," followed by the main obstacles. After it forms this summary, the server sends it to the phone along with the objects' data in JSON. The latency is a few hundred milliseconds, depending on the network and processing. In our tests, local Wi-Fi and an edge GPU gave 200–300 ms per cycle, which is enough for walking speed. This design is flexible. If there is no local server, we can connect to a cloud server with similar processing. The data can be encrypted for security. In this prototype, the server runs on a private local network. We can also upgrade any part without changing the rest. For example, if we have a more specialized indoor model, we can switch it in on the server side. The summarization logic can also be changed to make messages shorter or more detailed. We see EchoVision as a system that can improve over time, combining phone-based sensors with server-based AI.

IV. RESULTS OF EXPERIMENTS

We tested Echo Vision indoors and outdoors to check detection accuracy, response speed, and how well the guidance works. Indoor tests were in an office corridor and a room with chairs, tables, doors, and trash bins. Outdoor tests were on a residential street with sidewalks, parked cars, passing pedestrians, and common road-side objects.

Places365 recognized indoor or outdoor in over 95% of frames after a short delay. When moving from a building to the street, the system switched modes within about one second. This allowed the correct detection model to run most of the time. Sometimes, changes in lighting made it uncertain, but the system waited for a few frames to confirm before switching.

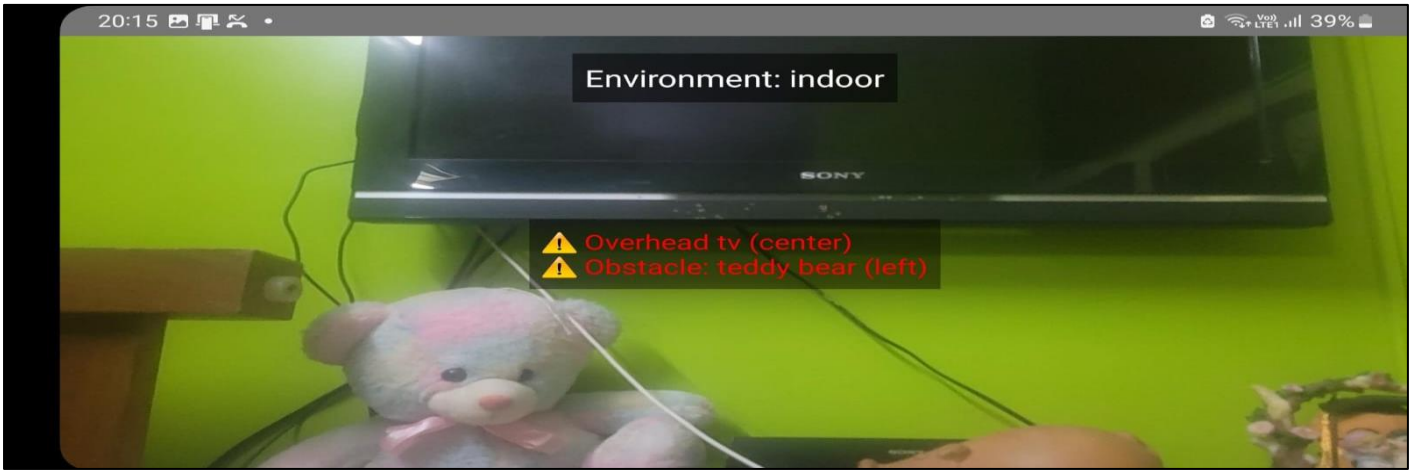

Fig 4 Overlay Testing Detection Example

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

Indoors, YOLOv8n detected items like chairs, desks, and doors reliably. Thin objects, like narrow poles, were sometimes missed, possibly due to COCO training data limitations. Outdoors, SSD MobileNet found people, cars, and other objects in view. For example, it detected a pedestrian at around 20 m ("Far") and tracked them until they passed close ("Near"), triggering vibration. It also detected parked cars and lampposts. These tests showed that switching models was helpful for focusing on objects that matter in each environment.

The simple distance estimation bins gave rough ranges that were found helpful. Vibration for objects under 1 m was useful outdoors. If a cyclist came close, the phone vibrated quickly, even before the audio could play. Indoors, the vibration alerted users to a chair blocking a path. Users did not get vibration for objects beyond 1 m.

The system gave voice messages about every 3 seconds when the user was moving. For example, indoors it would say, "Indoor mode. Ahead, the door at five meters," then later, "Doorway one meter ahead." Outdoors, it might say, "Outdoor mode. Vehicle on left, sidewalk clear ahead," and then, "Traffic light pole on your right at two meters." The time-based summaries were enough for understanding the scene without too many messages. In crowded areas, the summary listed several obstacles, making the audio longer. We plan to refine this to focus on the most important obstacles. Quick detections that vanished fast were usually not announced. This reduced unnecessary messages.

The end-to-end delay from capture to spoken feedback was about 150 ms indoors (YOLOv8n) and 120 ms outdoors (SSD) with local Wi-Fi and a GPU. We reached about 6–7 FPS when needed, but the frame rate often stayed around 2–3 FPS in quiet scenes with the load-aware throttle as seen in Fig 5. This saved half or more of the network traffic. If the network dropped briefly, we saw that the system should handle server unavailability better in the future

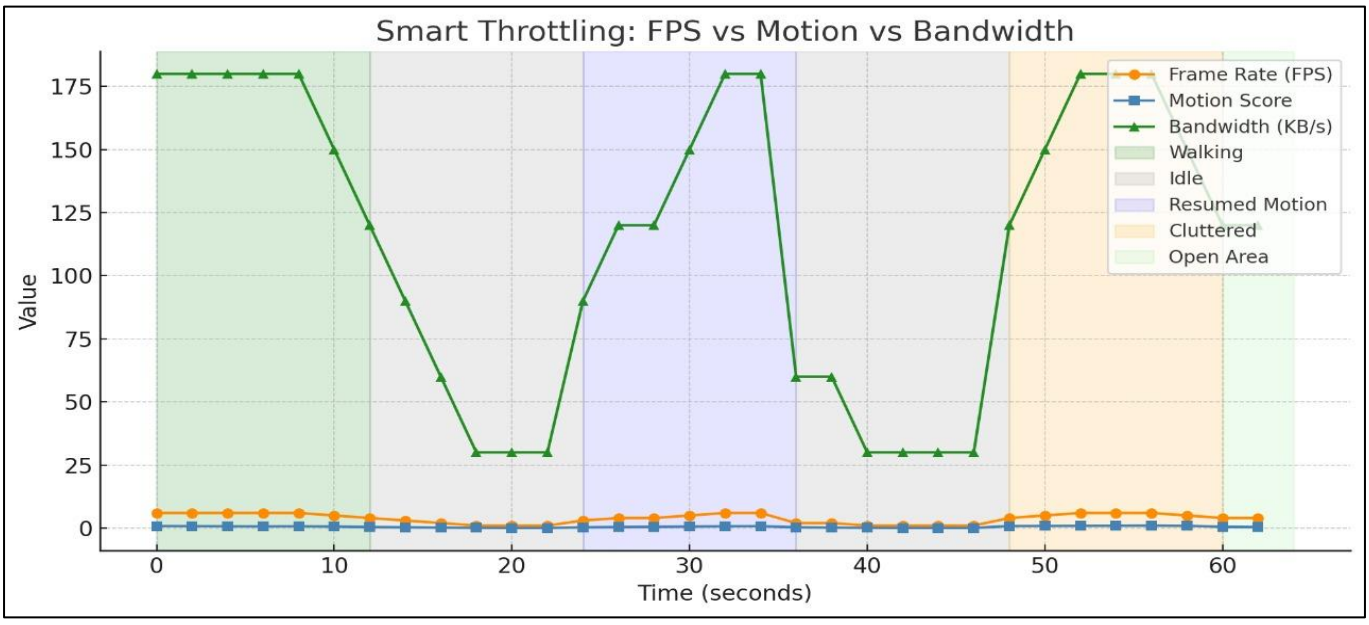

Fig 5 Smart FPS Throttle

Tracking ran smoothly during many tests. ARCore kept bounding boxes in place in the camera view. ORB-SLAM3 was enough for older phones, but it lost track in areas without clear features. The Flask server processed requests and kept the memory module within set limits. We will add features to notify the user if the connection to the server is lost.

Overall, tests showed that EchoVision can detect obstacles and adjust to indoor or outdoor conditions. By using the right model for each environment and summarizing detections over time, the system gives useful guidance without too many alerts. This supports ideas from other assistive devices that use context and intelligent filtering helps the user.

V. CONCLUSION & FUTURE WORKS

This paper explains EchoVision, a phone-based navigation aid for people with visual impairments that uses AR tracking, context-aware object detection, and offloading to a local server. The main feature is switching between indoor and outdoor detection pipelines by using a Places365 classifier. This allows EchoVision to choose YOLOv8 for indoor items or SSD-MobileNet for outdoor objects. ARCore helps with localization on newer devices, and ORB-SLAM3 is available as a backup.

By sending frames to a local server, we can run deep learning models that might not fit on a phone. The tests suggest that this setup works in real time, and it can be expanded in the future. EchoVision also uses a short memory to make speech guidance less repetitive. Multimodal feedback (visual overlays, voice, and vibration) can assist

ISSN No: -2456-2165 https://doi.org/10.38124/ijisrt/25sep880

people with different levels of vision.Our experiments show that EchoVision switches environment context and detects objects well, guiding users indoors and outdoors. In future work, we plan to include depth estimates from ARCore's depth API, test with more participants and try model optimization for times when the server connection is weak

We see EchoVision as a flexible platform for improving navigation for visually impaired users.

REFERENCES

[1]. D. Dakopoulos, "Tyflos: A Wearable Navigation Prototype for Blind & Visually Impaired; Design, Modelling and Experimental Results," Ph.D. dissertation, Dept. of Computer Science, Wright State University, Dayton, OH, 2009.

[2]. Y. H. Lee and G. Medioni, "RGB-D Camera Based Navigation for the Visually Impaired," Technical Report, Dept. of Electrical Engineering and Dept. of Computer Science, University of Southern California, Los Angeles, CA 90089, 2011.

[3]. A. Ben Atitallah, Y. Said, M. A. Ben Atitallah, M. Albekairi, K. Kaaniche, and S. Boubaker, "An effective obstacle detection system using deep learning advantages to aid blind and visually impaired navigation," Ain Shams Engineering Journal, vol. 15, no. 2, Art. no. 102387, 2024, doi: 10.1016/j.asej.2023.102387.

[4]. G. Jocher et al., "YOLO by Ultralytics (YOLOv8)," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[5]. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[6]. B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 Million Image Database for Scene Recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 6, pp. 1452–1464, Jun. 2018, doi: 10.1109/TPAMI.2017.2723009.

[7]. C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM," IEEE Trans. Robot., vol. 37, no. 6, pp. 1874–1890, Dec. 2021, doi: 10.1109/TRO.2021.3075644.

[8]. Google Inc., "ARCore: Augmented Reality for Android," [Online]. Available: https://developers.google.com/ar.

[9]. W. Liu et al., "SSD: Single Shot MultiBox Detector," in Computer Vision – ECCV 2016, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Lecture Notes in Computer Science, vol. 9905, Springer, Cham, Sep. 2016, doi: 10.1007/978-3-319-46448-0_2.

[10]. A. Howard et al., "Searching for MobileNetV3," in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 1314–1324.

[11]. H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," Comput. Vis. Image Understand., vol. 110, no. 3, pp. 346–359, 2008, doi: 10.1007/11744023_32.

[12]. D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," Int. J. Comput. Vis., vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.

[13]. M. H. Abidi, A. N. Siddiquee, H. Alkhalefah, and V. Srivastava, "A Comprehensive Review of Navigation Systems for Visually Impaired Individuals," Heliyon, Art. no. e31825, May 2024, doi: 10.1016/j.heliyon.2024.e31825.

[14]. M. Matei and L. Alboaie, "Enhancing Accessible Navigation: A Fusion of Speech, Gesture, and Sonification for the Visually Impaired," Procedia Computer Science, vol. 246, pp. 2558–2567, 2024, doi: 10.1016/j.procs.2024.09.442.

[15]. M. Aharchi and M. Ait Kbir, "Enhancing Navigation for the Visually Impaired Through Object Detection and 3D Audio Feedback," Journal of Theoretical and Applied Information Technology, vol. 102, no. 19, Article 6966, Oct. 15, 2024.

[16]. L. Manirajee et al., "Assistive Technology for Visually Impaired Individuals: A Systematic Literature Review (SLR)," Int. J. Acad. Res. Bus. Soc. Sci., vol. 14, no. 2, Feb. 2024, doi: 10.6007/IJARBSS/v14-i2/20827.