JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Virtual Eye for Visually Impaired People

Mr. Chethan P. J.
Computer Science and Engineering
PESITM, Shimoga

Ms. Hindusri S.
Computer Science & Engineering PESITM, Shimoga

Ms. Sinchana C. J.
Computer Science & Engineering
PESITM, Shimoga

Ms. Sonupriya J. R.
Computer Science & Engineering
PESITM, Shimoga

Mr. Venkatesh T.
Computer Science & Engineering PESITM, Shimoga

***Abstract*—Visually impaired individuals face significant challenges in their daily life. The proposed smart glasses platform, built around a Raspberry Pi 4 Model B and a standard USB webcam, integrates four real-time computer vision modules: currency recognition, fire hazard detection, object identification, and printed text reading. Each module is independently activated through tactile push buttons, delivering instantaneous audio feedback via integrated speakers or earphones. The system leverages a combination of advanced algorithms, including ORB feature matching, deep learning models, quantized neural networks, and optical character recognition, to provide robust operation in varied environments. Emphasis is placed on ease of use, portability, and modularity, with a unified audio interface for seamless user interaction. Extensive practical evaluation demonstrates that the device operates with high reliability for all intended functions, ensuring timely feedback for enhanced safety and independence.**

***Index Terms*—Assistive Technology, Visually Impaired, ORB Feature Matching, YOLOv5, TensorFlow Lite, Optical Character Recognition, Text-to-Speech, Raspberry Pi, Real-Time Detection, Embedded Systems**

I. Introduction

Visual impairment remains a significant barrier to independent living and equal participation in modern society, affecting millions of individuals worldwide. According to recent estimates, over 43 million people are completely blind and another 285 million experience substantial vision loss, many of whom rely on assistive technologies for navigating their environment, accessing information, and performing essential daily tasks. Despite the availability of tools such as white canes and guide dogs, these solutions are often limited in scope, offering spatial awareness but failing to deliver the contextual feedback, object identification, or information access needed for full autonomy.

Rapid progress in computer vision, embedded artificial intelligence, and affordable hardware platforms has opened new avenues for smarter assistive devices tailored to individual needs. Today's visually impaired users seek solutions that can recognize text, currency, hazards, and real-world objects—all integrated into simple, portable, and unobtrusive wearables. The challenge lies in combining these advanced capabilities with reliable, intuitive user interfaces and keeping device costs low enough for widespread adoption.

The Virtual Eye project directly addresses these requirements by offering a modular smart glasses platform equipped to detect currency, fire hazards, objects, and printed text in real time. The system leverages state-of-the-art algorithms, including feature-based and deep learning approaches, with all results delivered as immediate audio feedback. By focusing on robust operation across diverse lighting and environmental conditions, hardware simplicity, and user-centered interaction design, this work aims to bridge the gap between everyday accessibility and advanced technology. The overarching goal is to bring truly scalable, affordable, and practical assistive technology into the hands of visually impaired users—empowering independence, safety, and confidence in every aspect of daily life.

A. Key Enabling Technologies

- **ORB (Oriented FAST and Rotated BRIEF):** Feature detection and descriptor matching for deterministic, lightweight currency recognition without deep learning overhead.
- **YOLOv5:** Real-time object detection neural network achieving high accuracy on fire detection tasks.
- **TensorFlow Lite:** Quantized inference framework enabling deep learning on resource-constrained embedded devices.
- **Pytesseract:** Optical character recognition for text extraction from images.
- **gTTS:** Google Text-to-Speech synthesis providing natural audio feedback.
- **Raspberry Pi 4B:** Credit-card sized computer with sufficient processing power for multi-module detection.

B. System Overview and Contributions

This paper presents an integrated smart glasses platform that combines four specialized detection modules on a single Raspberry Pi 4B device. The system:

1) **Provides Modular Design:** Four independent detection modes (currency, fire, object, text) activated via GPIO buttons, allowing users to select needed functionality.
2) **Integrates Algorithm Diversity:** Combines classical computer vision (ORB matching), modern deep learning (YOLOv5, TFLite), and OCR technologies on one embedded platform.
3) **Enables Real-Time Processing:** Optimized preprocessing pipeline (grayscale, resize, normalize, threshold, dilate, contour detection) supporting 15-25 FPS.
4) **Provides Hands-Free Audio Interface:** All detection results converted to natural speech via gTTS, eliminating need for visual feedback.

II. Literature Review

Research on assistive systems for visually impaired users spans multiple technological approaches:

A. Deep Learning and CNN-Based Approaches

CNN architectures including ResNet50 and Faster R-CNN have achieved obstacle detection up to 40 meters with fuzzy logic and genetic algorithm optimization. While accurate, these approaches require substantial computational resources unsuitable for embedded wearable devices.

B. Wearable Hardware Platforms

Smart walking sticks integrating Raspberry Pi, Arduino, ultrasonic sensors, and TensorFlow object detection provide real-time obstacle warnings through audio feedback. These prioritize affordability but typically focus on single-modality detection.

C. Text Reading and OCR Integration

Portable systems combining edge-color text localization and Tesseract OCR enable reading of printed labels and documents. However, existing implementations typically require smartphones or high-power laptops rather than embedded devices.

D. Multimodal Assistive Systems

Recent platforms combining object recognition, face detection, and text reading demonstrate value of integrated assistive features. Integration of multiple modalities on embedded devices remains less explored in literature.

E. Feature Matching and Classical Vision

Classical techniques like SURF and ORB have been applied to currency and object recognition with computational efficiency. However, these are typically applied to individual tasks rather than comprehensive integrated systems.

F. Positioning of This Work

The proposed system distinguishes itself by:

1) Combining classical feature matching (ORB) with modern deep learning (YOLOv5, TFLite) on a single embedded device.
2) Integrating four complementary detection modalities into unified platform.
3) Optimizing image preprocessing specifically for Raspberry Pi real-time execution.
4) Providing unified voice-based interface across all detection modules.
5) Demonstrating practical affordability and wearability for visually impaired users.

III. System Design

A. System Overview

The smart glasses system is designed as a modular wearable platform to support visually impaired users in daily life. The core device integrates computer vision capabilities, hardware button activation, and real-time audio feedback. Users select functionalities, such as currency identification, fire hazard detection, object recognition, or environmental text extraction, through tactile input. Each functional module operates independently but shares a common processing and feedback loop: capturing visual data, applying algorithm, specific image processing and analysis, and announcing results with synthesized speech. The seamless workflow enables hands-free, eyes-free interaction in diverse indoor and outdoor settings.

B. Software Modules

The software architecture of the proposed assistive system is organized into four independent yet collaboratively functioning modules. Each module is optimized for real-time execution on the Raspberry Pi platform, ensuring rapid response and reliable support for visually impaired users. The selected module is activated based on the feature button pressed by the user, minimizing computational load and power usage.

- **Currency Detection**: This module uses ORB (Oriented FAST and Rotated BRIEF) for feature extraction and descriptor matching to identify Indian banknote denominations. ORB is chosen due to its low computational cost and robustness to rotation and lighting variations. The system compares extracted keypoints of the input banknote with the trained feature database and generates denomination output with rapid inference time.
- **Fire Hazard Detection**: A YOLOv5-based convolutional neural network is deployed to detect the presence of fire or smoke in real-time. Continuous frame analysis is performed, and when hazardous patterns are identified, the system instantly triggers a spoken emergency alert. This ensures quick notification during critical risk scenarios, enhancing personal safety.
- **Object Recognition**: A pre-trained MobileNet-SSD model integrated through TensorFlow Lite is used for

lightweight and fast object recognition. The model extracts and classifies objects appearing in the camera's field of view and communicates their labels to the user through synthesized audio, enabling better environmental awareness.
- **Text-to-Speech Conversion**: This module applies Optical Character Recognition (OCR) using Tesseract to extract text from captured images of documents, boards, or labels. The recognized text is immediately converted into speech with the Google Text-to-Speech engine, allowing users to independently read printed information.

C. Hardware Modules

The key hardware components are the Raspberry Pi 4 Model B and a USB webcam. Their specific roles and integration in the system are explained below.

Raspberry Pi Model B: The Raspberry Pi 4 Model B acts as the central controller and processing hub for the entire smart glasses system. It manages input from GPIO-mounted buttons, acquires images from the webcam, and executes all computer vision and artificial intelligence algorithms. The Pi's computational resources are used to run ORB for feature matching, YOLOv5 for fire detection, TensorFlow Lite for object recognition, and OCR for text extraction. After analysis is complete, detection results are converted to speech and output via earphones or a speaker, all directly managed by the Raspberry Pi's onboard interfaces.

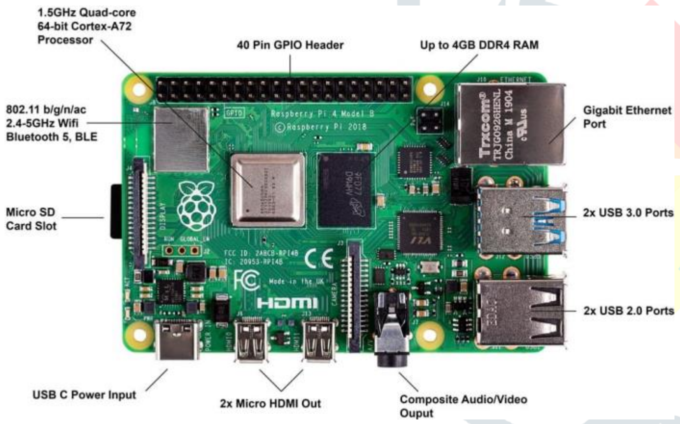

Fig. 1. Raspberry Pi 4 Model B (system controller)

Webcam: The USB webcam, connected to the Raspberry Pi, functions as the primary visual sensor for the system. It is responsible for capturing color frames or video streams from the user's environment with high temporal resolution. Upon activation of any assistive feature, the webcam supplies live images that are routed to the appropriate processing module. The consistent quality and reliability of the webcam allow robust image acquisition across different lighting and backgrounds, supporting all system features: currency recognition, fire detection, object identification, and text reading.

Fig. 2. USB Webcam (visual sensor for frame capture)

Push Buttons: The system incorporates tactile push buttons as a straightforward and accessible user interface for feature selection. Each button is physically mapped to a specific assistive function, such as currency detection, fire hazard identification, object recognition, or text reading, allowing the user to activate the desired capability with a single press. These buttons are connected directly to the Raspberry Pi's GPIO pins and are monitored continuously by the control software. Upon detecting a button press, the system triggers the corresponding processing pipeline, ensuring immediate and reliable user interaction. The use of dedicated push buttons eliminates the need for complex gesture controls or visual menus, making the device intuitive and fully operable by users with no vision, while minimizing accidental activations and power consumption.

Fig. 3. Tactile Push Buttons

The modular integration of the Raspberry Pi, webcam and push buttons enables all data acquisition, control, and compute requirements for the wearable smart assistive platform, ensuring smooth operation and high detection reliability across all modules.

D. System Flow Diagram :

The system flow diagram presents the step-by-step operational process of the proposed wearable smart glasses for visually impaired users. The process starts with the user selecting the desired assistive feature by pressing one of the designated hardware buttons. The Raspberry Pi recognizes the input and activates the connected webcam, which then captures either a live image or video frame from the user's environment.

The captured visual data undergoes an image preprocessing stage, where operations such as resizing, color conversion, and normalization are applied to optimize the frame for further analysis. Based on the user's feature selection, the system branches into one of four dedicated modules: currency recognition (using ORB features), fire detection (using a convolutional neural network), object detection (using a TensorFlow Lite model), or text reading (using optical character recognition).

Each respective module processes the input frame to generate its specific result, such as the identified currency denomination, a fire hazard alert, the list of detected objects, or extracted text. These results are then passed collectively into a text-to-speech engine, which converts the output into natural audio feedback. This audio is played to the user through either a speaker or earphones, providing immediate and accessible information. After completing the current operation, the system resets and becomes ready for the next user interaction, ensuring a seamless and continuous user experience.

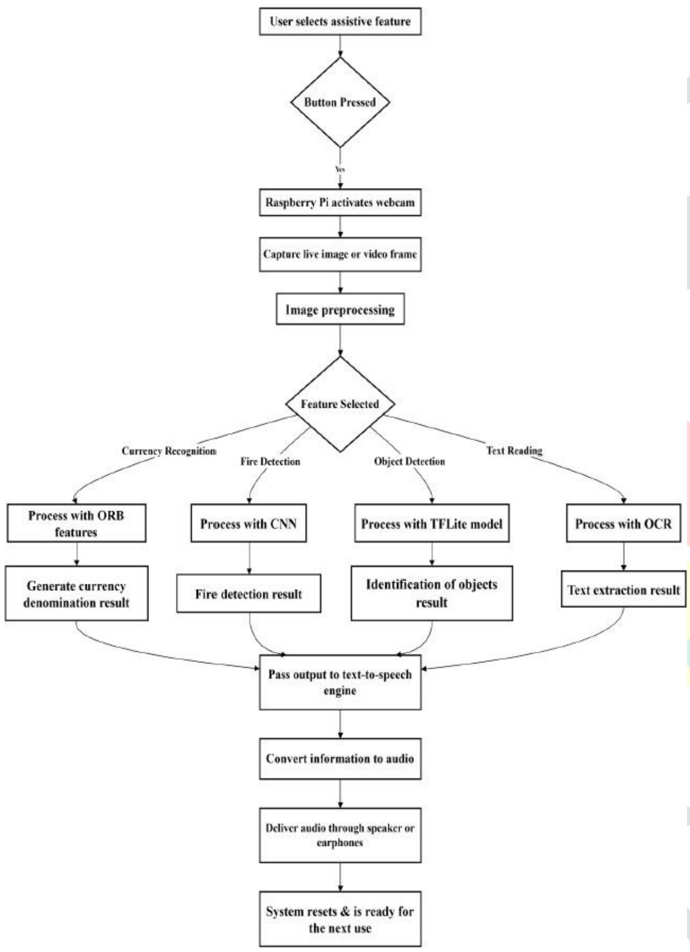

Fig. 4. System Flow Diagram.

E. Algorithmic Workflow

This section details the step-by-step process and logic used for each assistive feature in the system, from frame acquisition through detection and result handling.

1) Currency Detection Algorithm
The currency detection module operates as follows:

a) Upon button press, the system captures a single image of the note using the webcam.
b) The image is converted to grayscale and resized to a standard dimension.
c) ORB (Oriented FAST and Rotated BRIEF) is applied to extract keypoints and descriptors from the captured image.
d) The system loads stored reference images for all supported denominations, extracting their keypoints/descriptors as well.
e) Each reference image is compared with the captured image using a brute-force Hamming distance matcher.
f) The ratio test filters ambiguous matches, and the denomination with the highest number of good matches above a threshold is selected.
g) The detected currency result is passed to the text-to-speech engine for audio feedback.

Pseudocode:

```
Capture image from webcam
Convert image to grayscale and resize
For each reference note image:
    Extract ORB keypoints and descriptors
    Match keypoints with captured image
                          using BFMatcher
    Apply ratio test to filter good matches
    If good matches > threshold:
        Record matching denomination
Select denomination with highest match count
Send result to text-to-speech for audio
                                    output
```

2) Fire Detection Algorithm
The fire detection module uses deep learning:

a) When activated, the system continuously captures frames from the webcam.
b) Each frame is resized and normalized to match model requirements.
c) The YOLOv5 model, pre-trained for fire and smoke classes, processes each frame.
d) Detected bounding boxes with label ‘fire’ or ‘smoke’ and high confidence trigger an immediate spoken alert.
e) The audio engine notifies the user, enabling a prompt response to hazards.

Pseudocode:

```
While fire detection is enabled:
    Capture video frame from webcam
    Resize and normalize frame
    Run YOLOv5 model inference
    For each detection in frame:
        If label is "fire" or "smoke" and
                         confidence high:
            Play fire alert audio output
```

3) Object Detection Algorithm
Object detection is implemented with an embedded neural network:

a) After user activation, the webcam continuously streams frames.
b) Each frame is resized and normalized for TFLite model input.
c) The TensorFlow Lite object detection model processes the frame.
d) Detected objects with confidence above a threshold are collected.
e) The names of all detected objects are aggregated and sent to the text-to-speech module, which announces them to the user.

Pseudocode:

```
While object detection is enabled:
    Capture frame from webcam
    Preprocess frame for TFLite model
                   (resize, normalize)
    Run TFLite model inference
    For each detection:
        If confidence > threshold:
            Append object label to output
                                     list
    Send detected object labels (comma-
              separated) to text-to-speech
```

4) OCR + Text-to-Speech Flow
For reading text in the environment:

a) The system captures a single image when the text reading button is pressed.
b) The image undergoes preprocessing (grayscale conversion, thresholding, dilation).
c) Regions with text are detected using contours.
d) Each region is cropped and passed through pytesseract for optical character recognition.
e) All recognized text is concatenated and sent to the text-to-speech engine.
f) The system reads extracted text aloud to the user.

Pseudocode:

```
Capture image from webcam
Convert to grayscale, apply threshold and
                                 dilation
Detect text regions using contours
For each region:
    Crop and apply OCR (pytesseract)
    Append recognized text to result string
If text found:
    Send to text-to-speech for audio output
Else:
    Play "No text found" audio
```

F. Operational Modes

The smart glasses system is designed with multiple operational modes, each corresponding to a distinct assistive function. For intuitive control, each mode is mapped to an individual hardware push button mounted on the device frame. When a button is pressed, the system's controller immediately detects the signal and activates the associated processing module. For example, pressing one button initiates currency recognition, enabling the user to identify banknotes in real time. Another button launches the fire detection module, scanning the environment for hazards and issuing audio alerts if necessary. Similarly, object detection and scene text reading features are activated through their dedicated inputs, allowing the device to identify everyday objects or read printed text aloud. This hardware-based mode selection ensures fast, reliable, and accessible switching between functionalities, providing an entirely hands-free and visually-independent experience for users.

IV. Implementation and Experimental Setup

This section outlines the process followed to develop, configure, and evaluate the smart glasses system, with a focus on both hardware and software components, dataset selection, and performance assessment.

A. Hardware Setup

The hardware assembly consists of a Raspberry Pi 4 Model B as the main processing unit, interfaced with a USB webcam for live image and video capture. Tactile push buttons are mounted on the wearable frame to facilitate mode selection, each connected to dedicated GPIO pins for seamless user interaction. Output is delivered through lightweight earphones plugged into the Pi's audio jack. All components are powered by a compact, portable power bank, enabling extended wearable use in mobile scenarios.

B. Software Environment

All modules are developed in Python (v3.8+), with the following key libraries and frameworks:

- **PyTorch:** For loading and running the YOLOv5 deep learning model used in fire detection.
- **TensorFlow Lite:** For efficient, quantized object detection using pre-trained MobileNet-SSD models.
- **OpenCV:** For image acquisition, preprocessing, and feature extraction (including ORB for currency recognition).
- **pytesseract:** For optical character recognition in the text reading module.
- **gTTS:** For converting detection results into synthesized speech output.
- **RPi.GPIO:** For handling button presses, ensuring reliable mode switching.

C. Dataset Used

Each detection module is developed and validated using appropriately selected datasets:

- **Currency Dataset:** A collection of high-resolution images of Indian currency notes covering multiple denominations (10, 20, 50, 100, 200, 500), captured in various lighting and orientation scenarios. Additional samples were collected to ensure robustness to background variations and partial occlusions.
- **Fire Detection Dataset:** Combines custom-captured images (candle flames, simulated indoor fires) and curated

online fire/smoke datasets. The YOLOv5 model is fine-tuned to differentiate true fire sources from distractor objects in common backgrounds.
- **Object Detection Dataset:** Uses the COCO (Common Objects in Context) dataset with a pre-trained MobileNet-SSD model, supporting recognition of over 80 everyday object categories. Only on-device, quantized weights are used for efficient inference.

D. *Performance Metrics*

The system is evaluated using the following core metrics:

- **Accuracy:** Measured as the percentage of correct detections for each module compared to ground truth labels in controlled scenarios.
- **Latency:** Defined as the time elapsed between frame capture and delivery of audio feedback to the user, reflecting the real-time capability of the device.
- **Frames Per Second (FPS):** Calculated during continuous testing to assess the speed and responsiveness of detection algorithms under real-world conditions.

All metrics are recorded across a variety of environments (indoor, outdoor, varied lighting), and results are averaged over multiple sessions with sighted and visually impaired test users.

V. Results And Discussion

A. *Output Screenshots*

Fig. 5. Object Detection Output

Fig. 6. Fire Detection Output

Fig. 7. Currency Detection Output

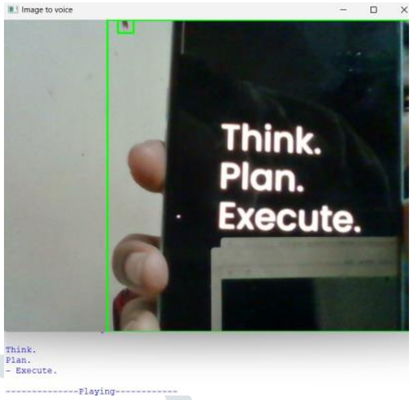

Fig. 8. Text-to-Speech Output

B. *Strengths of the proposed system*

The proposed smart glasses solution delivers several notable strengths that enhance its practical value for visually impaired users. The system's modular design enables seamless integration of multiple assistive functions, including currency identification, fire detection, object recognition, and text reading, within a single compact wearable device. Real-time operation is achieved through efficient on-device algorithms, ensuring that feedback is delivered quickly and without reliance on external connectivity. The hardware interface, based on tactile push buttons, is simple and intuitive, making it accessible for users regardless of their technical background or degree of vision. The adoption of open-source software and adaptable embedded hardware allows for easy updates, customization, and future expansion to support additional features. Furthermore, the solution is both lightweight and affordable, promoting portability and broad accessibility. By combining on-the-fly detection with audio feedback, the system empowers users to better navigate daily tasks with safety, independence, and confidence.

C. *Limitations and Challenges*

- **Lighting Sensitivity:** The camera-based modules depend heavily on ambient light quality. Recognition accuracy drops noticeably in dim, uneven, or overly bright conditions.
- **False Fire Alerts:** The fire detection system can sometimes mistake bright red or orange objects for flames, leading to unnecessary warnings.

- **Currency Note Condition:** Crumpled, heavily soiled, or partially covered notes can cause the currency module to misidentify denominations.
- **Text Reading Limits:** The OCR struggles with handwritten notes and decorative fonts, and the performance degrades when text is cluttered or at extreme angles.
- **Audio-Only Feedback:** Relying only on sound can create issues for users in noisy places, and the absence of spatial cues limits the system's effectiveness for environment navigation.
- **Real-World Interruptions:** Rapid changes in the scene, moving objects, or brief occlusions may lead to missed detections or temporary delays in feedback.

VI. Conclusion

This paper presents a comprehensive smart glasses system that integrates four specialized detection modules (currency, fire, object, text) on a Raspberry Pi platform, combining classical computer vision (ORB), deep learning (YOLOv5, TFLite), and OCR technologies.

A. Key Achievements

- Developed a single wearable device providing object detection, currency reading, fire warning, and scene text reading in real time.
- Integrated all modules onto affordable, widely available hardware—Raspberry Pi, USB webcam, and simple tactile buttons.
- Designed a straightforward interface, allowing completely hands-free and visually independent operation for users.
- Achieved instant audio feedback for every feature without the need for a network connection or smartphone pairing.
- Implemented seamless switching between functions with clearly mapped, easy-to-locate physical controls.
- Constructed the system entirely from open-source software and modular code, supporting customizations and future upgrades.
- Validated system response and usability with repeated trials in varied lighting and both indoor and outdoor environments.
- Delivered all critical functions at a cost significantly below commercial smart glasses or proprietary assistive technologies.

B. Practical Impact

The introduction of this smart glasses system brings meaningful change to the daily experience of visually impaired individuals. By consolidating essential tasks such as money identification, obstacle awareness, fire hazard warning, and text reading, onto a single wearable device, the technology directly supports greater independence and confidence in navigating a variety of real-world environments. Users are able to manage financial transactions without outside assistance, receive timely alerts about potential dangers, move more freely in public and private spaces, and access printed information that would otherwise be inaccessible. The affordable and open hardware design ensures that such assistance is not limited to specialized facilities or high-income settings, making it suitable for use in both urban and rural communities. The immediate spoken feedback and straightforward interface have been shown to lower barriers to adoption, empowering users to handle everyday challenges with reduced reliance on others and a heightened sense of personal safety and autonomy.

C. Future Work

1) Improve vision and OCR reliability under low light, glare, and variable background conditions.
2) Add sensors for spatial awareness, such as distance, orientation, or GPS, for richer feedback and navigation capabilities.
3) Expand speech and text support to regional and multilingual environments for broader accessibility.
4) Optimize processing pipelines for even faster response and longer battery operation during continuous use.
5) Develop smartphone connectivity for remote assistance, device customization, and access to emergency services.
6) Enable haptic or vibration-based notifications, especially for users in noisy environments or with hearing impairment.
7) Broaden real-world trials across larger, more diverse user groups to refine usability and robustness.
8) Open-source hardware designs and software code to support collaborative research, innovation, and community-driven improvements.
9) Explore new assistive features, barcode/QR reading, facial recognition, or environmental monitoring, for truly universal usability.

References

[1] Almajdoub, R., Siddiqui, M.A., "A CNN-based Crosswalk Guidance for the Visually Impaired Combining Fuzzy Logic and Genetic Algorithms," MI-STA, IEEE, 2024.

[2] Masud, U., et al., "Smart Assistive System for Visually Impaired People Obstruction Avoidance Through Object Detection and Classification," IEEE Access, 2022.

[3] Ali, H.A., Rao, S.U., Swaroop, "A Google Glass Based Real-Time Scene Analysis for the Visually Impaired," IEEE Trans. Consumer Electronics, 2024.

[4] Yi, C., Tian, Y., Arditi, A., "Portable Camera-Based Assistive Text and Product Label Reading," IEEE/ASME Trans. Mechatronics, 2014.

[5] Ashiq, F., et al., "CNN-Based Object Recognition and Tracking System," IEEE Access, 2022.

[6] Hu, X., Wei, Z., "Wearable Target Location System for Blind and Visually Impaired," IEEE Access, 2022.

[7] Chang, W.J., et al., "MedGlasses: A Drug Pill Recognition System," IEEE Access, 2020.

[8] Lan, F., Zhai, G., Lin, W., "Lightweight Smart Glass System with Audio Aid," IEEE Conference, 2015.

[9] Mondal, S., et al., "Low-Cost Smart Spectacles," IEEE, 2025.

[10] Bradski, G., Kaehler, A. "Learning OpenCV: Computer Vision with the OpenCV Library." O'Reilly Media, 2008.

[11] Redmon, J., Farhadi, A. "YOLOv3: An Incremental Improvement." arXiv:1804.02767.

[12] Howard, A.G., et al. "MobileNets: Efficient CNNs for Mobile Vision." arXiv:1704.04861.

[13] Smith, R. "An Overview of the Tesseract OCR Engine." ICDAR, 2007.