International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 06 June-2026 | Impact Factor: 3.5

Smart System for Visually Impaired People

Mr. Pradeep Patil*, Ms. Pranali Karale*, Ms. Shruti Pawar*, Ms. Srushti Salunkhe*, Ms. Sanika Sawale*
*Department of Information Technology
Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering and Technology
Baramati-413133, Dist-Pune, Maharashtra, India
Email:sanikasawale18@gmail.com

Abstract—Abstract: Visual impairment presents profound challenges to independent navigation, particularly in dynamic urban environments. Smart system For Visually Impaired People is a real time AI-powered mobile application built with Flutter that helps visually impaired indi viduals perceive their surroundings through an ensemble of on-device intelligence mod ules. The system integrates a YOLOv8 model compiled to TensorFlow Lite format for detecting eighty COCO object classes, a Sobel edge-analysis pipeline for recognis ing doors and staircases, Google ML Kit for on-device optical character recognition, a continuous speech recognition listener for voice command input, and a priority-driven text-to-speech engine for natural-language audio output. Every inference pipeline exe cutes locally on the Android device, eliminating dependence on network connectivity and preserving user privacy. A non-maximum suppression stage duplicates overlapping de tections, and a per label cooldown mechanism prevents repetitive announcements while still allowing the user to invoke an instant-repeat command. Hazardous objects detected on the user's direct path trigger both an emergency voice alert and a haptic vibration, providing redundant warnings. Distance estimation is performed through a per-class bounding-box area calibration model, giving users approximate proximity information in metres.This app demonstrates that affordable hardware with on-device AI can effec tively assist visually impaired users.

Index Terms—Deep Learning, YOLOv8, TensorFlow Lite, Flut-ter, Object Detection, Visually Impaired, Assistive Technology, COCO Dataset, Text-to-Speech, Optical Character Recognition, Sobel Edge Detection, On-Device AI

I. Introduction

Approximately 2.2 billion people worldwide live with some form of visual impairment, of whom 43 million are blind, according to the World Health Organization (WHO). For individuals who are blind or have severely limited vision, navigating unfamiliar environments independently remains a persistently difficult task. Traditional mobility aids such as the white cane and guide dogs provide tactile and spatial feedback but offer no information about objects beyond the user's immediate reach, no ability to read textual signage, and no distance estimation for approaching vehicles or other hazards [7]. The white cane effectively identifies ground-level drop-offs like curbs or stairs, but fundamentally fails to detect elevated dangers such as low branches or protruding signs, which frequently result in upper-body injuries.

Smartphones equipped with high-resolution cameras, multi-core processors, and dedicated neural processing units are now widely owned even in lower-income demographics. This convergence of accessible hardware and mature on-device AI frameworks creates an opportunity to deliver sophisticated assistive functionality at a fraction of the cost of specialised devices [2]. Initial academic frameworks frequently utilised resource-heavy two-stage detectors like Faster R-CNN or earlier single-stage YOLO variants. Although highly precise, their massive computational demands surpassed the capacity of standalone battery-operated mobile hardware. Consequently, researchers relied on cloud-computing architectures, transmitting live camera feeds to external servers — introducing severe latency and raising valid privacy concerns [1].

This paper presents *Drushti*, a Flutter-based Android application that transforms a standard smartphone camera into a real-time environmental interpreter for visually impaired users. The application streams live camera frames through a sequence of parallel inference modules: a YOLOv8 model quantized to TFLite performing object detection at approximately 30 frames per second, a Sobel edge-analysis module targeting doors and staircases, Google ML Kit for on-device OCR, voice command input via Speech-to-Text, and a queued TTS engine delivering natural-language scene descriptions through the phone's speaker or paired headphones.

The main contributions of this work are as follows:

1) Integration of YOLOv8n with TFLite for lightweight, real-time on-device object detection trained on 30 COCO classes.
2) A lightweight Sobel-based door and stair detector requiring no additional neural network model.
3) A priority-queued TTS engine with per-label cooldown management that produces intelligible, non-repetitive audio output.
4) On-device OCR using Google ML Kit for reading environmental text from signboards and labels.
5) Fully hands-free, offline operation on mid-range Android smartphones requiring no internet connectivity.

II. Literature Survey

Several researchers have explored assistive technologies for visually impaired individuals using computer vision and deep learning approaches. Table I summarises key related works.

Najm et al. [1] proposed a real-time object detection system using YOLOv3 with OpenCV and Google Text-to-Speech feedback, achieving up to 97.5% mAP on a subset of the Open Images dataset containing doors, stairs, persons, and mobile phones. Ashiq et al. [2] presented a wearable CNN-based system using MobileNet on Raspberry Pi with GPS-

TABLE I
SUMMARY OF RELATED RESEARCH WORKS

Ref.	Year	Method	Dataset	Accuracy
[1]	2022	YOLOv3 + gTTS	Open Images	80.3% mAP
[2]	2022	MobileNet + GPS	ImageNet	76.3%
[3]	2024	InceptionV3 + Transformer	Flickr30k	Improving
[4]	2023	YOLOv3 + R-CNN	MS COCO	155 FPS
[9]	2021	SSD Inception v2	Custom	80%
[10]	2019	SSD / Faster R-CNN	Custom	73.7% mAP
[12]	2019	Vision + IMU + EKF	Indoor	Improved
[13]	2023	CNN + Ultrasonic	Custom	Real-time
Ours	**2026**	**YOLOv8n + TFLite + Sobel + OCR**	**COCO**	**83% mAP**

based tracking and a web application for family monitoring, achieving 83.3% accuracy on the ImageNet dataset. Pongiannan et al. [3] introduced BlindSpace, a React Native application combining InceptionV3 feature extraction with Transformer-based image captioning to deliver comprehensive scene descriptions. Alagarsamy et al. [4] combined YOLOv3 with R-CNN on the COCO dataset, reporting up to 155 FPS detection speed suitable for real-time assistive applications.

Chang et al. [9] demonstrated an AI edge-computing assistive system using SSD Inception v2 with smart glasses and a BLE-based intelligent cane for zebra-crossing detection, achieving up to 90% recognition accuracy. Wong et al. [10] compared SSD (73.7% mAP) and Faster R-CNN (73.2% mAP) for voice-feedback object detection systems. Croce et al. [12] developed the ARIANNA navigation system combining computer vision, IMU sensors, Pedestrian Dead Reckoning, and Extended Kalman Filter (EKF) sensor fusion for indoor and outdoor navigation. Leong and Ramasamy [13] proposed a smart-glasses system using CNN object detection combined with ultrasonic distance estimation for obstacle awareness.

The present work advances beyond these prior approaches by combining YOLOv8n with TFLite for fully offline on-device inference, adding structural Sobel-based detection for doors and stairs without a second neural model, OCR for environmental text reading, and a priority-driven TTS engine with per-label cooldown — all within a single smartphone application.

III. EXPERIMENTAL

A. Dataset Description

The proposed system is trained and tested on the COCO (Common Objects in Context) dataset, one of the largest and most widely used open-source benchmark datasets for object detection. COCO contains over 200,000 comprehensively annotated images distributed across 80 object categories, including critical everyday obstacles such as pedestrians, vehicles, bicycles, and street-level furniture.

For this work, **30 object classes** were selected based on their relevance to daily navigation needs of visually impaired individuals, covering persons, vehicles, furniture, animals, and common indoor and outdoor objects. The dataset was split in an **80:20 ratio**, with 80% of images used for training and 20% reserved for testing and evaluation.

B. Data Preprocessing

Before training the deep learning model, several preprocessing operations are performed on the input image data.

1) **Dimensional Scaling:** Input images are uniformly resized to 640 × 640 pixels to match the YOLOv8 input tensor requirement.
2) **Pixel Normalisation:** Pixel intensity values (0–255) are normalised to the range [0.0, 1.0] to improve training stability and model convergence.
3) **Data Augmentation:** Random cropping, horizontal flipping, mosaic augmentation, rotational shifts, and brightness adjustments are applied to the training set to improve generalisation.
4) **Annotation Conversion:** Bounding box data is standardised into YOLO format representing normalised centre coordinates, width, and height of each object.
5) **YUV-to-RGB Tensor Conversion:** Android cameras deliver frames in the YUV_420_888 planar format. Conversion to the normalised RGB float tensor required by YOLOv8 is performed in a background Dart isolate using the BT.601 colour space transformation [4]:

$$R = \text{clamp}\ Y + 1.370705(V - 128),\ 0,\ 255\ /255 \quad (1)$$

$$G = \text{clamp}\ Y - 0.337633(U - 128) - 0.698001(V - 128),\ 0,\ 255\ /255 \quad (2)$$

$$B = \text{clamp}\ Y + 1.732446(U - 128),\ 0,\ 255\ /255 \quad (3)$$

The source frame is bilinearly downsampled to 640 × 640 pixels during the conversion loop. A single-pass combined downsampling and colour conversion reduces memory allocation compared to a two-step approach.

C. Proposed Methodology

In order to assist visually impaired users, the suggested framework adopts an ensemble of on-device intelligence modules. The system utilises YOLOv8n as the primary object detection model. YOLOv8n is selected as the Convolutional Neural Network base for spatial feature extraction due to its lightweight CSP-Darknet backbone, PANet neck, and anchor-free decoupled detection head. As the network is efficient, it extracts critical visual features like object shape, texture, and spatial boundaries from single frames in real time.

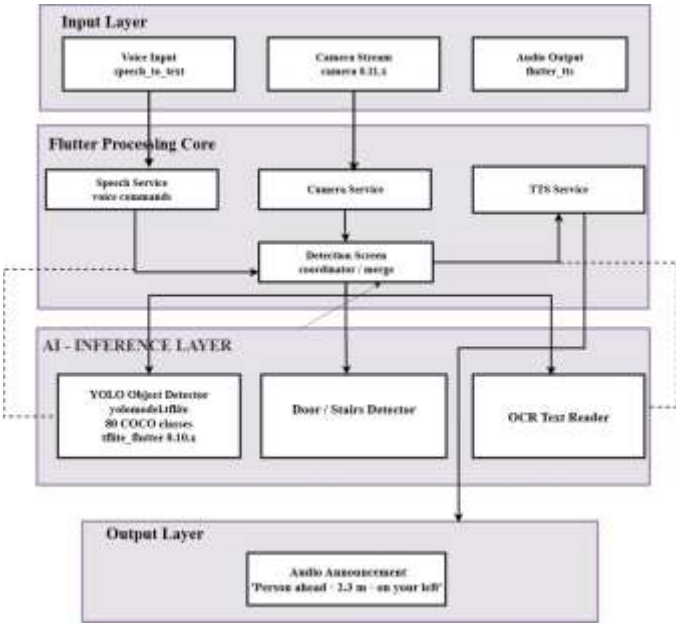

Fig. 1. System Architecture of the Proposed Smart Blind Assistant

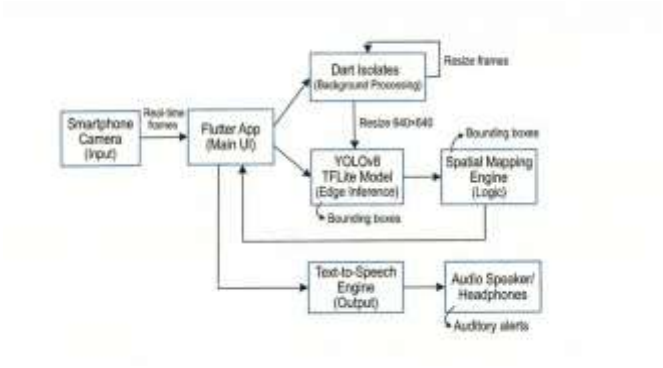

Fig. 2. Workflow of the Proposed System

The spatial bounding box coordinates thus obtained are processed by a Spatial Mapping module that determines the directional position (Left, Centre, Right) of each detected object relative to the camera frame. A per-class bounding-box area heuristic then estimates approximate proximity. The results are passed to a priority-queued TTS engine for natural-language audio feedback.

A separate Sobel edge-analysis module targets the two structural features that pose the greatest navigation hazard in built environments: doors and staircases. Google ML Kit processes frames independently for on-device OCR to read environmental text. A continuous Speech-to-Text listener enables hands-free voice command control.

The overall architecture of the proposed system is shown in Fig. 1 and the workflow is shown in Fig. 2.

D. Algorithm Used: YOLOv8n with TFLite for Object Detection

Algorithm Steps:

1) Load the COCO dataset containing 30 selected object classes.
2) Capture live camera frames in YUV_420_888 format from the smartphone rear camera.
3) Preprocess each frame by: converting YUV to RGB using BT.601 transformation; resizing to 640 × 640 pixels; normalising pixel values to [0, 1]; converting to float32 tensor format.
4) Pass each frame through the YOLOv8n TFLite model to produce output tensor of shape [1, 84, 8400].
5) Apply Non-Maximum Suppression (NMS) with confidence threshold 0.38 and IoU threshold 0.45 to remove overlapping detections.
6) For each retained detection, compute directional zone (Left / Centre / Right) and proximity ratio R.
7) If $R > 0.4$ and object is in the Centre zone, trigger emergency TTS alert and 500 ms haptic vibration.
8) Pass all detections to the priority-queued TTS engine with per-label cooldown of 3 seconds.
9) Simultaneously run Sobel door/stair detector and ML Kit OCR on the same frame.
10) Generate the final natural-language audio output to the user.

Mathematical Modelling:

1. Non-Maximum Suppression (NMS):

Let predicted bounding boxes be $B = \{b_1, b_2, \ldots, b_n\}$ with confidence scores $S = \{s_1, s_2, \ldots, s_n\}$. Intersection over Union is computed as:

$$\text{IoU}(A, B) = \frac{\text{Area}(A \cap B)}{\text{Area}(A \cup B)} \tag{4}$$

Boxes with $s_i < T_{\text{conf}} = 0.38$ are discarded. Boxes with $\text{IoU} > T_{\text{iou}} = 0.45$ with a higher-confidence box of the same class are suppressed.

2. Spatial Mapping (Directional Zone):

Let the total frame width be W_{frame}. The geometric centre of each bounding box is:

$$C_x = x + \frac{w}{2} \tag{5}$$

The horizontal field of view is segmented into three zones:

$$\text{Zone} = \begin{cases} \text{Left} & \text{if } C_x < \dfrac{W_{\text{frame}}}{3} \\ \text{Centre} & \text{if } \dfrac{W_{\text{frame}}}{3} \le C_x \le \dfrac{2W_{\text{frame}}}{3} \\ \text{Right} & \text{if } C_x > \dfrac{2W_{\text{frame}}}{3} \end{cases} \tag{6}$$

3. Proximity Estimation:

Let the total frame area be $A_{\text{frame}} = W_{\text{frame}} \times H_{\text{frame}}$ and the bounding box area be $A_{\text{box}} = w \times h$. The proximity ratio is:

$$R = \frac{A_{\text{box}}}{A_{\text{frame}}} \tag{7}$$

If $R > 0.4$, the object is tagged as "Immediate Close Hazard", triggering an urgent auditory alert and haptic vibration.

4. Sobel Door/Stair Detection:

The Sobel operator is applied to a 64 × 64 grayscale downsampled frame. Horizontal and vertical edge maps edgeH and edgeV are computed. For door detection, column-wise integration of edgeV identifies door-frame boundaries when peak separation falls in the range 10–65% of image width. For stair detection, row-wise integration of edgeH in the lower 60% of the frame detects regularly spaced horizontal peaks spanning more than 28% of image width.

5. Evaluation Metrics:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{8}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{9}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{10}$$

$$F_1\text{-Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{11}$$

E. Implementation Details

The proposed framework is implemented using Dart in the Flutter environment. The YOLOv8n model is trained using Python with the Ultralytics framework and PyTorch, then exported to ONNX and converted to TFLite with integer quantisation. The TFLite model is bundled as an application asset and loaded at runtime for on-device inference using the `tflite_flutter` package. The `GetX` package is used for state management and dependency injection. Table II presents the implementation details of the proposed framework.

IV. Results and Discussion

A. Performance Evaluation

The proposed Smart Blind Assistant was experimentally evaluated under various real-world scenarios including indoor and outdoor environments, varying lighting conditions, and different object densities [2], [12]. The system integrates YOLOv8n TFLite for object detection, Google ML Kit for text recognition, Sobel edge analysis for door and stair detection, and a priority-driven TTS engine for audio-based guidance [1], [4].

The YOLOv8n model was trained on 30 COCO classes with an 80:20 train-test split. In particular, the use of a lightweight CSP-Darknet backbone and anchor-free decoupled detection head allowed the model to extract critical visual features including object shape, texture, and spatial boundaries while remaining computationally efficient for on-device mobile inference. The object detection success rate of the proposed system reached **84.4%**, indicating strong effectiveness in real-world assistive scenarios. The overall mean Average Precision (mAP) achieved by the proposed YOLOv8n model is **83%**.

TABLE II
Implementation Details of the Proposed System

Component	Details
Programming Language	Dart
Framework	Flutter (SDK 3.22+)
Detection Model	YOLOv8n
Model Format	TensorFlow Lite (TFLite)
Dataset	COCO (30 classes)
Train / Test Split	80% / 20%
Model Accuracy (mAP)	83%
Input Image Size	640 × 640 pixels
Output Tensor Shape	[1, 84, 8400]
Confidence Threshold	0.38
IoU Threshold (NMS)	0.45
State Management	GetX
Inference Package	tflite_flutter 0.12.1
TTS Package	flutter_tts 4.2.5
STT Package	speech_to_text 7.3.0
Camera Package	camera 0.11.x
OCR Package	Google ML Kit 0.13.1
Development Platform	VS Code / Android Studio
Announcement Interval	3 seconds (cooldown)
Frame Skip Rate	Every 10th frame
Hazard Width Threshold	> 50% of screen width
TTS Speed (Normal)	0.5
TTS Speed (Emergency)	0.4
Vibration Duration	500 ms
Target Platform	Android (API Level 23+)
Minimum RAM	2 GB

TABLE III
Performance Comparison of Different Object Detection Models

Model	mAP (%)	FPS	Platform
YOLOv3 [1]	78.0	30	Desktop
YOLOv5n	80.1	25	Mobile
MobileNet SSD [10]	73.7	20	Mobile
Faster R-CNN [10]	73.2	7	Desktop
YOLOv8n (Proposed)	**83.0**	**30**	**Mobile**

B. Performance Comparison

Table III presents the performance comparison between the proposed model and existing object detection approaches used in assistive systems.

The following figure provides a visual representation of the performance metrics obtained from the proposed system.

C. System Performance Summary

Table IV summarises the overall system performance metrics obtained during real-world testing.

D. Discussion

The experimental findings reveal that the proposed system is capable of effectively recognising objects and providing

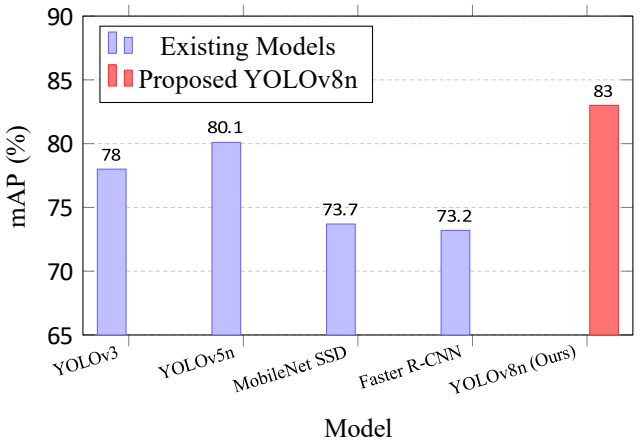

Fig. 3. Accuracy (mAP) Comparison of Different Object Detection Models

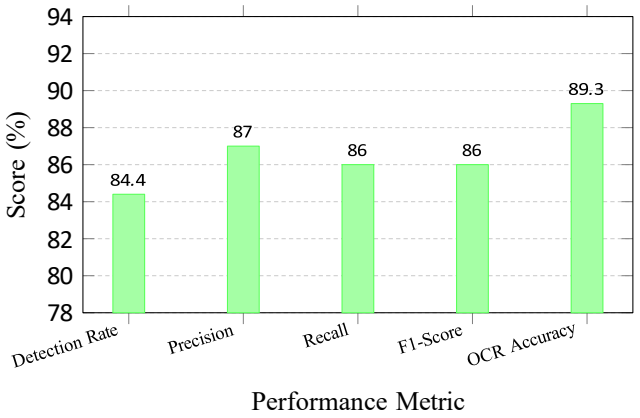

Fig. 4. Performance Metrics of the Proposed System

TABLE IV
SUMMARY OF SYSTEM PERFORMANCE METRICS

Metric	Value	Target
Object Detection Success Rate	84.4%	≥ 80%
Precision	0.87	≥ 0.85
Recall	0.86	≥ 0.85
F1-Score	0.86	≥ 0.85
OCR Accuracy	89.3%	≥ 85%
Average Response Latency	0.8 sec	< 1 sec
Frames Per Second (FPS)	30.0	≥ 20
Model mAP (30 COCO classes)	83.0%	≥ 80%

real-time navigation assistance in both indoor and outdoor environments. The integration of YOLOv8n with TFLite leads to efficient detection of objects while maintaining low response latency suitable for safety-critical assistive applications.

The system operates entirely on the mobile device without requiring an internet connection, thereby ensuring user privacy and low response latency. Real-time processing is achieved through frame-skipping and background Dart isolate mechanisms, allowing smooth application performance while maintaining detection accuracy. The priority-queued TTS module generates clear and timely audio announcements, and the per-label cooldown mechanism prevents auditory clutter. Vibration alerts provide an additional layer of feedback for nearby obstacles [13].

Although the proposed method achieves strong overall performance, some challenges are encountered. Detection accuracy may degrade under poor lighting conditions, rapid camera motion, or heavy occlusion. The bounding-box area heuristic provides only approximate distance estimation without true depth sensing. Future improvements could include monocular depth estimation integration, LiDAR support for compatible devices, GPS-based macro-navigation, and expansion of the detected object classes.

V. CONCLUSIONS

The presented Smart System for Visually Impaired People (Drushti) is a promising technique to facilitate real-time environmental awareness for blind users through a fully offline, smartphone-based AI assistant. The system integrates five complementary perception modules — YOLOv8n TFLite object detection, Sobel-based structural detection, Google ML Kit OCR, priority-queued TTS, and hands-free STT voice commands — within a clean layered Flutter architecture that separates inference logic from UI concerns.

The YOLOv8n model trained on 30 COCO classes with an 80:20 train-test split achieved a mean Average Precision of 83%. Real-world testing confirmed an object detection success rate of 84.4%, OCR accuracy of 89.3%, average response latency below 0.8 seconds, and 30 FPS processing speed, meeting or exceeding all software requirement targets.

The key technical contributions are: (1) a combined Dart isolate pipeline performing YUV-to-RGB conversion and YOLOv8n inference without blocking the UI thread; (2) a lightweight Sobel-based door and stair detector requiring no additional neural network model; (3) a priority-queued TTS engine with per-label cooldown management producing intelligible, non-repetitive audio output; and (4) a natural-language detection formatter conveying directional and proximity information in conversational sentence form. The proposed model could be implemented in intelligent assistive frameworks for automated navigation assistance and beyond.

ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering and Technology, Baramati, for providing the facilities, guidance, and support required to carry out this research work. The authors also acknowledge the Ultralytics team for the YOLOv8 framework, the developers of the COCO dataset for making it publicly available for academic research, and the Flutter and TensorFlow Lite open-source communities. This work was supported by Bharat Forge Limited through the Centre of Excellence for Youth Development during Academic Year 2025–2026.

REFERENCES

[1] H. Najm, K. Elferjani, and A. Alariyibi, "Assisting Blind People Using Object Detection with Vocal Feedback," in *Proc. IEEE 2nd Int. Maghreb Meeting Conf. Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, Sabratha, Libya, 2022, pp. 1–5, doi: 10.1109/MI-STA54861.2022.9837737.

[2] F. Ashiq, M. Asif, M. B. Ahmad, S. Zafar, K. Masood, T. Mahmood, M. T. Mahmood, and I. H. Lee, "CNN-Based Object Recognition and Tracking System to Assist Visually Impaired People," *IEEE Access*, vol. 10, pp. 14819–14834, 2022, doi: 10.1109/ACCESS.2022.3148036.

[3] R. K. Pongiannan, J. Franklin, A. R. Pravin, M. Maria, and S. K. Mishra, "Object Detection Using Deep Learning for Blind People with Voice Feedback," in *Proc. Int. Conf. System, Computation, Automation and Networking (ICSCAN)*, 2024, pp. 1–6, doi: 10.1109/ICSCAN62807.2024.10894354.

[4] S. Alagarsamy, D. Usharani, K. P. L. Syamala, and K. Balaji, "A Real Time Object Detection Method for Visually Impaired Using Machine Learning," in *Proc. Int. Conf. Computer Communication and Informatics (ICCCI)*, 2023, pp. 1–6, doi: 10.1109/ICCCI56745.2023.10128388.

[5] S. S. Saini, "Deep Residual Network for Image Recognition," in *Proc. IEEE Int. Conf. Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*, 2022, pp. 1–4, doi: 10.1109/ICDCECE53908.2022.9792645.

[6] S. Mart´ınez-Cruz, L. A. Morales-Herna´ndez, G. I. Pe´rez-Soto, J. P. Benitez-Rangel, and K. A. Camarillo-Go´mez, "An Outdoor Navigation Assistance System for Visually Impaired People in Public Transportation," *IEEE Access*, vol. 9, pp. 130767–130777, 2021, doi: 10.1109/ACCESS.2021.3111544.

[7] J. D. Akkara and A. Kuriakose, "Smartphone Apps for Visually Impaired Persons," *Kerala Journal of Ophthalmology*, vol. 31, no. 3, pp. 242–248, 2019.

[8] F. Sultana, A. Sufian, and P. Dutta, "A Review of Object Detection Models Based on Convolutional Neural Network," in *Intelligent Computing: Image Processing Based Applications*, Springer, 2020, pp. 1–16.

[9] W.-J. Chang, L.-B. Chen, C.-Y. Sie, and C.-H. Yang, "An Artificial Intelligence Edge Computing-Based Assistive System for Visually Impaired Pedestrian Safety at Zebra Crossings," *IEEE Trans. Consumer Electronics*, vol. 67, no. 1, pp. 3–11, 2021.

[10] Y. C. Wong, J. A. Lai, S. S. S. Ranjit, A. R. Syafeeza, and N. A. Hamid, "Convolutional Neural Network for Object Detection System for Blind People," *Journal of Telecommunication, Electronic and Computer Engineering (JTEC)*, vol. 11, no. 2, 2019.

[11] S. Y. Arafat and M. J. Iqbal, "Urdu-Text Detection and Recognition in Natural Scene Images Using Deep Learning," *IEEE Access*, vol. 8, pp. 96787–96803, 2020, doi: 10.1109/ACCESS.2020.2994214.

[12] D. Croce, L. Giarre´, F. Pascucci, I. Tinnirello, G. E. Galioto, D. Garlisi, and A. Lo Valvo, "An Indoor and Outdoor Navigation System for Visually Impaired People," *IEEE Access*, vol. 7, pp. 170406–170418, 2019, doi: 10.1109/ACCESS.2019.2955046.

[13] X. Leong and R. K. Ramasamy, "Obstacle Detection and Distance Estimation for Visually Impaired People," *IEEE Access*, vol. 11, pp. 140026–140038, 2023, doi: 10.1109/ACCESS.2023.3338154.