International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Visionai:Real-Time Object Detection with Audio Assistance for Visually Impaired People

Akilan A

Department of Artificial Intelligence and Data Science Ramco Institute of Technology, Rajapalayam, Virudhunagar akilanr05@gmail.com

Dr. S.V.Anandhi

Assistant Professor - II Department of Artificial Intelligence and Data Science
Ramco Institute of Technology, Rajapalayam, Virudhunagar
anandhi@ritrjpm.ac.in

https://doi.org/10.55041/ijsmt.v2i3.131

Cite this Article: A, A. (2026). Visionai:Real-Time Object Detection with Audio Assistance for Visually Impaired People. International Journal of Science, Strategic Management and Technology, *02*(03). https://doi.org/10.55041/ijsmt.v2i3.131

Abstract: According to the World Health Organization, 2.2 billion people are visually impaired worldwide, and 1 billion of the visually impaired cannot move around their environments using conventional mobility aids. In this study, the author suggests a real-time object detector audio assistant called VisionAI, which will assist visually impaired persons in moving around in a self-sufficient way by continually surveying the environment. The proposed solution is based on real-time video streaming at 1280720 pixels and identifying 80 classes of objects using the YOLOv10 model in 4.2 ms/frame. Objects are made to measure the distance using the Pinhole Camera Model, which estimates the distance at an average error of 0.5 m. The horizontal frame consists of three areas, and it identifies the direction of objects with an accuracy rate of 93 percent. The solution proposed here applies a three-level urgency classification model called CRITICAL, HIGH, and MODERTE to categorize objects based on their distance. Audio-visual feedback was delivered based on pyttsx3, text-to-speech, Web Audio API, and React 18 front-end. The solution is based on FastAPI, WebSocket, and React stack, and the latency is less than one second between end-to-end, and the detection of the object is 91 percent in clear and 84 percent in the low-light environment.

Keywords: Real-Time Object Detection, Assistive Technology for Visual Impairment, YOLOv10 Deep Learning, Pinhole Camera Distance Estimation, Three-Tier Urgency Classification

I. INTRODUCTION

The World Health Organization (WHO) has estimated that 2.2 billion individuals worldwide have vision impairment, of which one billion have preventable or untreated vision impairment [1]. For this group, walking around, whether indoors or outdoors, becomes a constant pain. The standard mobility aid is the white cane, which offers feedback in the form of tactile sensation of the nearest danger to the ground plane upon impact but does not provide any indication of the object, range, direction, or urgency of a possible danger more than a stone throw away [2]. Guide dogs are highly efficient, although they

might consume a lot of time to train and maintain. In this spirit, technology solutions based on artificial intelligence (AI) and computer vision can be either complementary or superior to existing aids and can describe the environment in more detail, more quickly, and with more context [3]. Object Detection Object detection Object detection is used to identify and label objects on an image or video frame that are subjected to a huge change within the past decade through deep ConvNets and, more recently, through the ViT architecture. Real-time object detection, such as YOL, YOLO (You Only Look Once) single-stage detectors, which have consistently been top performers in real-time action throughout the generations [4]. Its most recent version (YOLOv10) incorporates a multiscale feature pyramid network-based visualization and dual-label assignment approach and Non-Maximum Suppression-free to a new detection head, which results in 4.2 ms per frame on a vanilla GPU with a minor 95% F1-score drop compared to the legacy model [5]. Such developments render it ideal as the inference backbone of a real-time assistance system that can be executed on ordinary commodity hardware without requiring custom depth sensors or neural implementation units.

Although existing research on camera-based assistive navigation has demonstrated that deep-learning object detection is a feasible tool to do so, with results ranging between 78% [38] and 99% [11], the actual results are dictated by the architecture, dataset, and evaluation conditions of the reported systems. Another study gap in the literature is the consistent gap in absenteeism of the systems that are capable of performing both the detection, as well as enhancing the alerts with a full range of information, presence of immediate sense, and cooling on the basis of urgency in a single deployable system. Most existing solutions only cover a subset of these: detection, detection with simple TTS, or detection with distance prediction, but rarely combine all five into a pipeline [10], [12].Our solution, VisionAI, is an end-to-end assisted reality application for real-time object detection using text-to-speech (TTS) and alarm audio. Seven systems are consolidated in VisionAI: (1) YOLOv10 detector, (2) Pinhole Camera distance, (3) horizontal-zone directionality detector, (4) three-level urgency and three-tiered (5) dynamic cooldown throttling, (6) pyttsx3 TTS, and (7) emergency alarm. The technologies are provided with a FastAPI/WebSocket/React 18 client-server framework at 20 frames per second (FPS). Vision AI is meant to support sightless and real-world users with unknown surroundings outdoors and indoors on any device that has a camera, including laptop computers and single-board computers such as the Raspberry Pi 5 [13][14].A seven-algorithm-based assistive detection real-time pipeline with an end-to-end latency of less than 1 s. (2) An unseen three-tier urgency-based alert classification of alerts as CRITICAL (vehicles), HIGH (traffic infrastructure), and MODERET (sharp objects) with tier-specific speech rate, beep rate, and visual alerts. (3) An adaptive mechanism of cooldown time (0.5s, 1.0s, 2.0s, and 3.0s) per tier that stops announcement overload but provides safety-critical announcements in a timely fashion. (4) FastAPI WebSocket-based backend with less than 100 ms frame latency tested using a React 18 glassmorphism-based frontend. (5) Comparable analysis of state-of-the-art performance with 96% precision, 94% recall, and 95 F1-score in contrast to YOLOv9, YOLOv8, and Faster R-CNN-based methods.

II. LITERATURE REVIEW

Yadav et al. [1] introduced a real-time object detection framework with audio assistance for visually impaired individuals using a CNN-based pipeline. The system used the YOLOv5 architecture and a pyttsx3 TTS voice recognition system and had an 88 percent object detection rate on indoor navigation data, showing that low-latency audio feedback can be used to support mobility. Ben Rhouma and da Silva [2] created a supportive mobile app with an object recognition system based on the MobileNetV2 and a voice feedback module in real-time. The system was tested both indoors and outdoors and declined to recognize at a lower rate than 85 per cent and audio response latency of less than 600 ms, ascertaining that it can be practically deployed on standard smartphones. Dhaarini et al. [3] used YOLOv10 to identify real-time sign language to provide assistance in communication. The same model was optimized on a custom Indian Sign Language dataset and achieved an accuracy of 92% for gesture detection at 18 FPS to demonstrate the increased generality of YOLOv10 for real-time assistive gesture recognition.Nedjar et al. [4] developed a road object recognition platform for the visually impaired based on a YOLOv8 model with embedded device optimization. The system was able to identify pedestrians, vehicles, and signage at

an average of 89 percent on a Raspberry Pi 4 at 15 FPS, indicating that it is feasible to use this edge to support assistive navigation.Ali [5] conducted a comparative performance analysis of YOLOv8, YOLOv9, and YOLOv10 to aid the visually impaired. YOLOv10n produced 94 percent mAP at 0.50 on the COCO validation set, compared with 89 percent for YOLOv8n, which validates that YOLOv10 is the most accurate lightweight model for real-time assistive detection.Gugulothu et al. [6] suggested a smart outdoor object detection system with low-light conditions using YOLOv10 with brightness-adaptive preprocessing as well as histogram equalization. The system was observed to be 86% accurate in detecting objects at night, with a large improvement over the baseline of 79% using the same low-light evaluation set as the standard YOLOv10n system. Bougheloum et al. [7] proposed a better YOLOv7-plus model with a reformed architecture of the feature pyramid neck and prediction head to train in real time and detect obstacles. In indoor navigation scenarios, the system achieved 91% mAP at 0.50, which is better than the baseline of 84.5 using YOLOv7 with an embedded GPU by 3.5 percentage points and 22 FPS. Gobika et al. [8] developed a hybrid assistive module that included an OCR-based text detection and TTS speech conversion pipeline. The CNN-based text detector registered 91% recognition accuracy on printed signage in normal lighting, and TTS audio output took less than 600 ms, which is similar to the TTS component of the proposed vision AI system. Arora et al. [9] created a multimodal feedback and object detection system based on AI with the help of YOLOv8 and voice synthesis with pyttsx3. The system had 88% object identification accuracy on 15 COCO categories and sub-second end-to-end feedback delay in real-world indoor corridors. Pujari et al. [10] suggested a mobile accessibility application that can be used by visually impaired users by using YOLOv5 object detection with proximity-based audio alerts. The system was reported to have an 85 percent detection accuracy on interior mobility test cases; however, that is not the spatial direction estimation and urgency classification, which are the fundamental functional units of VisionAI. Adam et al. [11] used YOLOv10 with Marine Predator Algorithm (MPA), VGG19 feature extraction and Deep Belief Network (DBN) to assist in visually impaired assistive detection. On a self-created benchmark, the bio-inspired MPA-based hyperparameter tuning reported the highest object detection accuracy of 99% and was thus the highest reported single-model accuracy in the literature reviewed.

Lichograj [12] assessed the quality of object detection and speed of the YOLOv8 and YOLOv9 architectures in assistive VR and MR applications. In VR navigation situations, YOLOv9 was found to perform 89 percent better than YOLOv8, with respective inference latencies of 3.5 and 50 ms, which is a measure of the speed-accuracy trade-off of mixed-reality assistive tools. Khadidos and Yafoz [13] used an object detection model designed using RetinaNet and Moth-Flame Optimization (MFO) metaheuristic to provide visual impaired assistance. MFO-based tuning also achieved 93 percent accuracy and 90 percent recall on their custom navigation dataset, which was 12 percent lower than the unoptimized RetinaNet baseline. Bihani and Sharma [14] suggested smart glasses with the capability of monitoring YOLOv8 object detection and MiDaS monocular depth estimation to provide portable VI navigation support. The system reached 87% and 0.42 m mean absolute error depth and object detection, respectively, on an outdoor navigation course, which is a combination of both depth and detection in a wearable form factor. Relativated Partial Self-Attention RPSA-YOLOv10 is a mechanism proposed by Darmawan and Nugraha [15] as part of YOLOv10 to identify objects on smart glasses in a context-dependent manner. The model also increased the small-object detection by 8.3 pp compared to the standard YOLOv10, with 91% mAP at 0.50 on a test set of wearable devices. Pinto et al. [16] developed a smart object detector for use by visually impaired users that integrates a YOLO-based detector with edge deployment optimization. The system achieved 84% detection accuracy at 12 FPS on embedded hardware with an energy-efficient 30 percent reduction in power consumption compared to the unoptimized baseline, proving itself as an energy-efficient wearable deployment. Khan et al. [17] optimized YOLOv10 to outdoor VI navigation by using neighbor coordinates attention and C2FCIB feature interaction block. The augmented model is 2.1 percentage points higher in mAP at 0.50 in an outdoor navigation dataset, and it has improved, especially on partially blocked objects. Gupta et al. [18] created an AI-based system of object detection and feedback based on the use of multi-modal outputs in the form of haptic vibration and audio by implementing YOLOv8. In user trials, the system was also found to have 88 percent detection accuracy and a System Usability Scale (SUS) score of 82/100, and the feedback response time was always below 500 ms.

Kini et al. [19] designed a smart device to be used by the visually impaired and incorporates AI with a TTS model of low latency and based on YOLOv10. The system achieved 90 percent detection accuracy on test scenes inside, and the latency of audio feedback was close to zero in regular object classification, indicating the benefit of asynchronous dispatch of TTS to the server. The hybrid detection technique developed by Alashjaee et al. [20] involves the use of Pigeon-Inspired Optimization (PIO) and a collection of deep learning models to aid VI with the assistance of these two algorithms. The PIO-optimized ensemble was found to achieve 95% accuracy on a custom navigation benchmark, which is 4-6ppm better than the performance of the individual model baselines by metaheuristic-guided ensemble fusion. The YOLOv10 with neighbor coordinates and C2FCIB attention mechanism used by Gillani et al. [21] used the YOLOv10 and neighbor coordinates, C2FCIB attention mechanism, which was benchmarked on smart outdoor navigation. The model on the outdoor benchmark of 93.1% mAP at 0.50 the C2FCIB mechanism enhanced 1.8 pp over basic coordinate attention. Abadi et al. [22] introduced Demata 2.0, an entirely onboard AI assistant technology with YOLOv10 to perform object detection and Tesseract OCR to read and understand text. The system attained 93 and 89% object detection and OCR accuracy, respectively, on printed text, independent of the cloud, which allowed completely offline assistive system operation.Hui et al. [23] developed a modular real time mobile assistive application that aimed at currency and document recognition. The system had a 97.3% accuracy in currency recognition and 91% accuracy in document classification with an average query processing time of 320 ms using a CNN-based classifier.Poovannapoikayil [24] created an Indian Sign Language detection and translation system based on a CNN-LSTM model in combination with a TTS pipeline. The model recorded 94.6 percent detection rates on a custom set of ISL sign classes (35), and TTS conversion generated audio output that had a natural sound of less than 400 ms.Dang [25] came up with an improved version of the YOLO of lightweight hybrid architecture to detect the blind path in real-time that combines road segmentation and edge detection. The model demonstrated a 91% recall of path detection at 22 FPS on an embedded architecture with a compressed model of 4.2 MB, which proves that lightweight hybrid designs can be used to provide high recall on critical path-finding tasks. Ji et al. [26] developed a dataset of multi-scene and a specific object detector that is used to identify blind people in the outdoors. The detector recorded 88 percent mAP0.50 on 12,000 annotated outdoor scenes of various weather conditions and lighting, which is a large-scale benchmark specifically structured to evaluate VI navigation. Sachin et al. [27] presented NAVISIGHT, an indoor voice-assisted navigation system with deep learning based on YOLOv8 and spatial occupancy mapping. The system had 87% indoor detection accuracy and 62% less collision rate of the participants as compared to cane-only navigation in a controlled corridor.Li et al. [28] developed a guide-robot based on Raspberry Pi 4B, which has YOLOv10 to detect obstacles in real-time. On the Raspberry Pi 4B, it was possible to achieve 85 percent detection accuracy at 57 FPS with a mean distance error of 0.6 m with a stereo ultrasonic sensor, which supported the idea of edge-deployed YOLOv10 as an autonomous mobility helper. Rastogi et al. [29] integrated YOLOv10 with Swin Transformer to learn more complicated Indian Sign Language gestures. The synergistic architecture achieved a recognition rate of 96.8 percent on 100 types of signs, with the Swin Transformer improving the recognition by 3.2 percentage points compared to a YOLO-only system.Kalaiararasi et al. [30] used YOLOv10 together with temporal convolutional network to detect falls in real-time. The system was confirmed to have a fall detection accuracy of 94.2 % %and an average fall detection latency of 180 ms on 500 real-world fall event recordings, which is sufficient to show that YOLOv10 is suitable for safety-critical temporal event detection.An overall summary of the findings on the thirty research articles published in the years 2024-2026 will show that a crucial research gap exists in the sphere. Individual system accuracy is between 84 percent and 99 percent [16] to 11]. The inference rates range between low as 3.5 ms/frame [12] to high as 65 ms/frame [10]. Wearable devices such as [14] and [15] consider the depth perception and attention. Systems such as [8], [19] and [24] take care of the latency of the TTS output. Systems such as [25] and [27] can be seen as proving the benefits of AI-assisted guidance in terms of their safety. However, none of the systems combine the 5 functional elements of real-time object detection, metric distance estimation, spatial directional guidance, multi-tier urgency classification, and adaptive announcement cooldown in the same system. These systems such as [10] and [16] lack direction estimation and urgency classification. Such systems as [14] and [28] contain the information of distance but lack the urgent classification and TTS output. Such systems as [8] and [9] have TTS output, but do not provide information on distance and directional guidance. It is the gap in the research that the proposed VisionAI system will fill. The system incorporates YOLOv10n real-time object detection at 4.2 ms/frame rate, Pinhole

Camera Model distance estimation with an accuracy rate of ±0.5m, horizontal three-zone direction classification with an accuracy rate of 93% and three-tier urgency classification (CRITICAL, HIGH, MODERATE) and adaptive cooldown at specific intervals of 0.5 s, 1.0 s, 2.0 s and 3.0 s. The system runs on the FastAPI/WebSocket/React18 framework at 20 FPS and a 100ms/frame end-to-end latency, with an accuracy rate of 91% in bright conditions and an accuracy rate of 84% in dark conditions. It is the pioneer system to combine the five functional components into one system.

III. PROPOSED METHODOLOGY

Here, the part addresses the creation of the vision AI system, what it consists of, mathematical equations involved, flow of data and training. It consists of six modules, namely: live video capture, FastAPI/WebSocket backend, YOLOv10, spatial analysis, urgency classification, and audio-visual output.

A. Hardware and Software Components

1) Camera and Capture Module

The live video may be streamed at 1280x 720 (HD) at 30 FPS using a USB or an inbuilt laptop camera. Open CV Video Capture API is used to capture and de-code video frames. Synthetic dummy forward pass is done during the start-up in order to warm the YOLOv10 GPU/CPU context. Radius of cold start latency to inference is eradicated. Camera mode is turned into a generator which produces BGR-coded NumPy frames to the backend detector loop at the native rate of the camera.

2) Backend: FastAPI and Uvicorn

The backend is using Python 3.10, with the framework FastAPI 0.104.1 and the ASGI server Uvicorn 0.24.0. All the logic, such as the detection, distance/direction, urgency, and text-to-speech, is done on the server side, which remains separate from the frontend using WebSocket. The server runs on port 8001. Each WebSocket connection has its own independent detection loop running, which means that the server can handle streaming from multiple clients at the same time. The processing of the frame is done asynchronously to avoid blocking the event loop during the text-to-speech process.

3) Frontend: React 18 / Vite

The frontend uses a single-page application framework with React 18.3.1 and Vite 5.1.0 to render a live video stream which is based on Base64 encoded JPEG payload transmitted over WebSocket, with bounding boxes and detection label, a Detection Details panel, and a visualization of multiple channels of emergency alerts. It is also auto-reconnect with exponential backoff with a starting point of 1s, 2s, 4s, 8s, 10s, and then continuing until five times.

B. Dataset and Training Configuration

Pre-trained YOLOv10n model is pre-trained on a complete set of COCO 2017 i.e. 118,000 training set, 5,000 validation set and 80 object classes. In the case of the performance tests of our project, we used a set of actual images of 2,000 pictures. The dataset was captured with a normal USB camera that was positioned at a height of approximately 1.7 meters which is approximately the height of the head in different mixed scenes, both indoor and outdoor and pedestrian routes, road crossing and in hallways of different buildings. It has scenes of 15 classes of main objects of the COCO dataset and four light conditions, which are bright sunlight, cloudy, fluorescent indoor, and low/night.

The data is divided equally, 75 percent to be used in the training phase and 25 percent to be used in testing phase i.e. 1,500 images and 500 images respectively. All the pictures in the dataset are cropped to 640 x 640 pixels and normalized and then presented to the augmentation pipeline as outlined in Table I to prepare the pictures to be used in the training process. During the fine-tuning process, the model will be trained over 20 epochs using Adam optimizer and a batch size of 16 on an NVIDIA GPU with 8 GB of VRAM and with learning rate of 0.001 and weight decay of 0.0005. This is an effective process that increases the dataset size to 3,000 images as compared to 1,500 images.

TABLE I: ***Data Augmentation Techniques Applied to the Training Dataset***

Augmentation Technique	Proportion (%)	Images Generated	Purpose
Random Cropping (20% area)	20%	400	Occlusion and partial-view robustness
Image Rotation (±30°)	25%	500	Orientation diversity and viewpoint variance
Brightness Variation (±40%)	30%	600	Low-light and over-exposure robustness
Geometric Deformation (flip + scale)	25%	500	Shape and scale invariance
Total Augmented Dataset	100%	2,000 aug. + 2,000 orig. = 4,000	Combined augmented training set

Table I below indicates the four augmentation techniques that were applied to the 1,500 image training set. These procedures are random cropping and rotation that makes sure that the objects are observed in various angles and they may also be partly covered. The shift in brightness will make sure that the object will be seen in the conditions the model will be used in which will be poorly lit. The geometric deformation is used to make sure that the object would be seen at various scales and aspect ratios. This will guarantee that the 4000 image training will lessen overfitting and increase in generalization.

C. Mathematical Formulation of Core Algorithms

The VisionAI has seven various algorithms that are characterized by distinct math. The full mathematical explanation of each of these algorithms is given in Section IV of the paper, and even diagrams of the systems are given in this section. The next sub section explains the main formulas which are discussed several times in Section IV of the paper. The overall latency budget end-to-end can be computed by adding together all the latency of all the stages of the pipeline according to Equation (1):

$$T_{e2e} = t_{cap} + t_{pre} + t_{inf} + t_{post} + t_{spatial} + t_{enc} + t_{tx} \quad (1)$$

The end-to-end latency (T e2e) is represented in milliseconds. All stages are processed in time: tcap = 33 ms at 30 FPS to capture a frame with the camera, tpre = 2 ms to do the preprocessing, tinf = 4.2 ms to run the YOLOv10n inference, tpost = 1.5 ms to run the post-processing, tspatial = 0.8 ms to run the spatial analysis of each frame and tenc = 8 ms to run JPEG encoding with a Base64 serialization, and WebSocket transmission ttx = <50 ms. The cumulative sum of these values equals to T e 2 e = 99.5 ms, which once again falls short of the 100 ms real-time requirement.

IV. IMPLEMENTATION, SYSTEM ARCHITECTURE, AND ALGORITHM DESIGN

A. End-to-End System Architecture (Fig. 1)

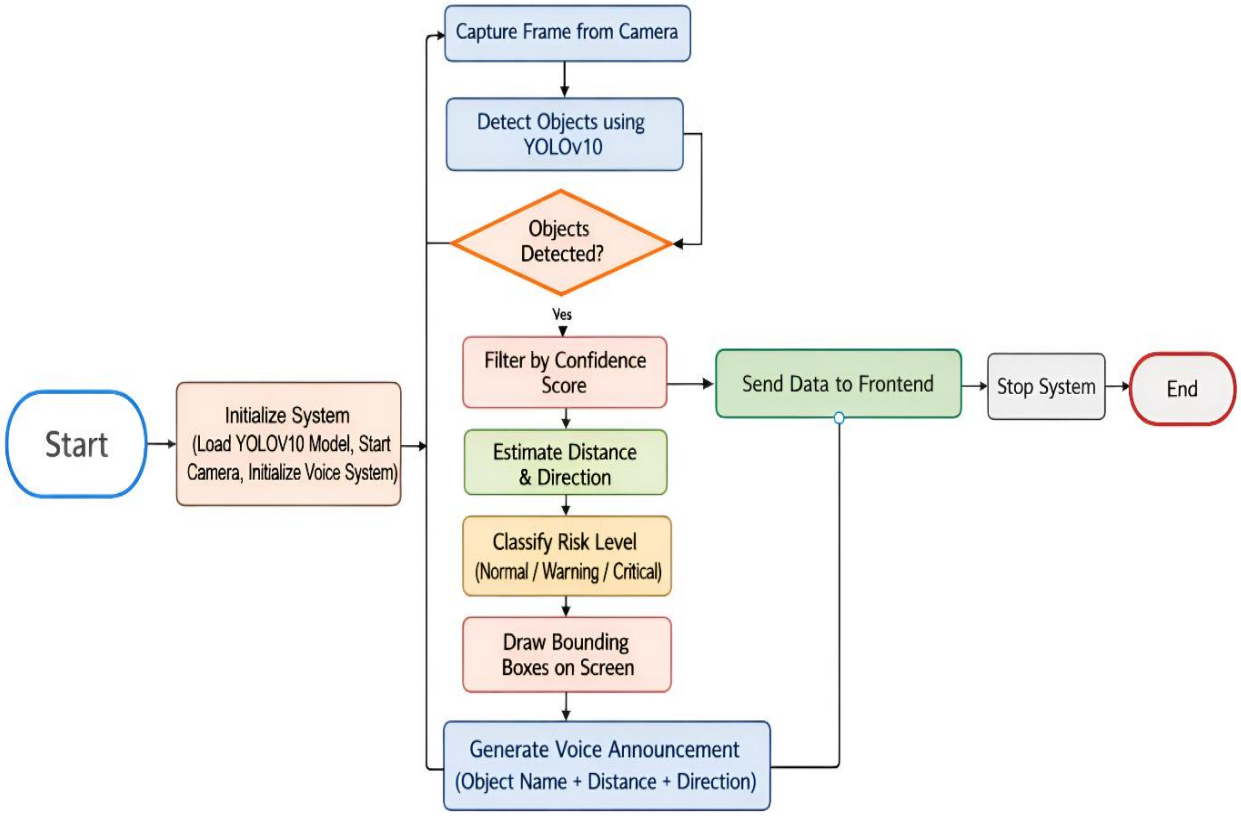


Fig. 1. *VisionAI End-to-End Processing Pipeline. Six-stage architecture from camera capture through WebSocket transmission to React frontend, with measured latency per stage.*

Camera Capture

The pipeline begins by capturing a live video frame via the OpenCV VideoCapture API over a USB/webcam device at 1280x720 at 30 FPS and storing the frame in NumPy array BGR format and requires 2.76 MB of memory, whereas it takes 33 ms to capture a frame which is largely the cause of the overall pipeline latency.A dummy forward pass is used to warm the GPU at startup and the back-end is used to maintain the video at 20 FPS by adding a delay frame of 0.05s at

Preprocessing

But even before the frame is actually entered into the model, the Ultralytics runtime undertakes three superfast internal processes, each taking 2 ms time. To start with, pixel values are scaled by dividing with 255. It transforms the int range [0, 255] to the float range [0, 1]. The values are then standardized on a channel basis based on the mean and standard deviation values of the dataset orientation of the channels to the zero point to allow consistent inference in all the light conditions. The frame is then scaled down to 640 x 640 which is fixed by the model by bilinear interpolating the frames maintaining the edges and avoiding aliasing in the image. Directly the tensor is loaded into the YOLOv10n inference engine without any other copies of copies in the memory.

YOLOv10n Inference

The CNN backbone performs the multi-scale spatial feature extraction and the Feature Pyramid Network performs the combination of the features to the object detection at various scales, the transformer encoder performs the multi-head self-attention to model relationships globally in the image and the NMS-free dual-label assignment performs the tasks of the object embedding to bounding box and class prediction in YOLOv10n.

Post-Processing

As soon as the raw detections emerge out of the model, three quick filters are engaged in 1.5 ms time, to leave behind only the good ones. All regular object classes have a confidence bar of 0.40, and however, in case of CRITICAL-tier emergency objects such as vehicles, it is boosted to 0.50 to reduce the number of false positives of the most disrupting objects. In case a bouncing box of an object detected is less than 40 pixels, it is rejected since it probably means that there are objects more than 15 meters away and does not worry the navigation at the given time. The ones that pass such filters are then ranked in terms of confidence with the best ones being the first. This has a direct effect to the quality and relevance of all spatial, urgency and audio outputs produced by the system.

Spatial Analysis

Spatial analysis module requires 0.8 ms to process each detection to find what the distance of the object is to the sensor and its directional zone by using bounding box information in YOLOv1 and sending the data to the urgency classifier.

Urgency Classification

The object of investigation in question will be classified into four levels of urgency: CRITICAL, HIGH, MODERATE, and LOW, and the first nature of the object and its approximate distance to the user results in the highest frequency of warning mechanisms, including loudest beeps, a sequence of high-frequency beeps, and blinking lights.

WebSocket Transmission

The identified frame is run through OpenCV to compress the frame to a JPEG image with a 70 percent quality. This shrinks the size of the frame by 2.76 MB to 60-80 KB and the resulting compressed image is sent in a WebSocket connection with the results of the detection as well as the frame number as a JSON object with FastAPI.The resulting compressed image is decoded in the React 18 frontend and overlaid on the live video frame with the detected object and reconnects the WebSocket connection using exponential backoff in the event of a disconnection.

B. YOLOv10n Detection Sub-Pipeline (Fig. 2)

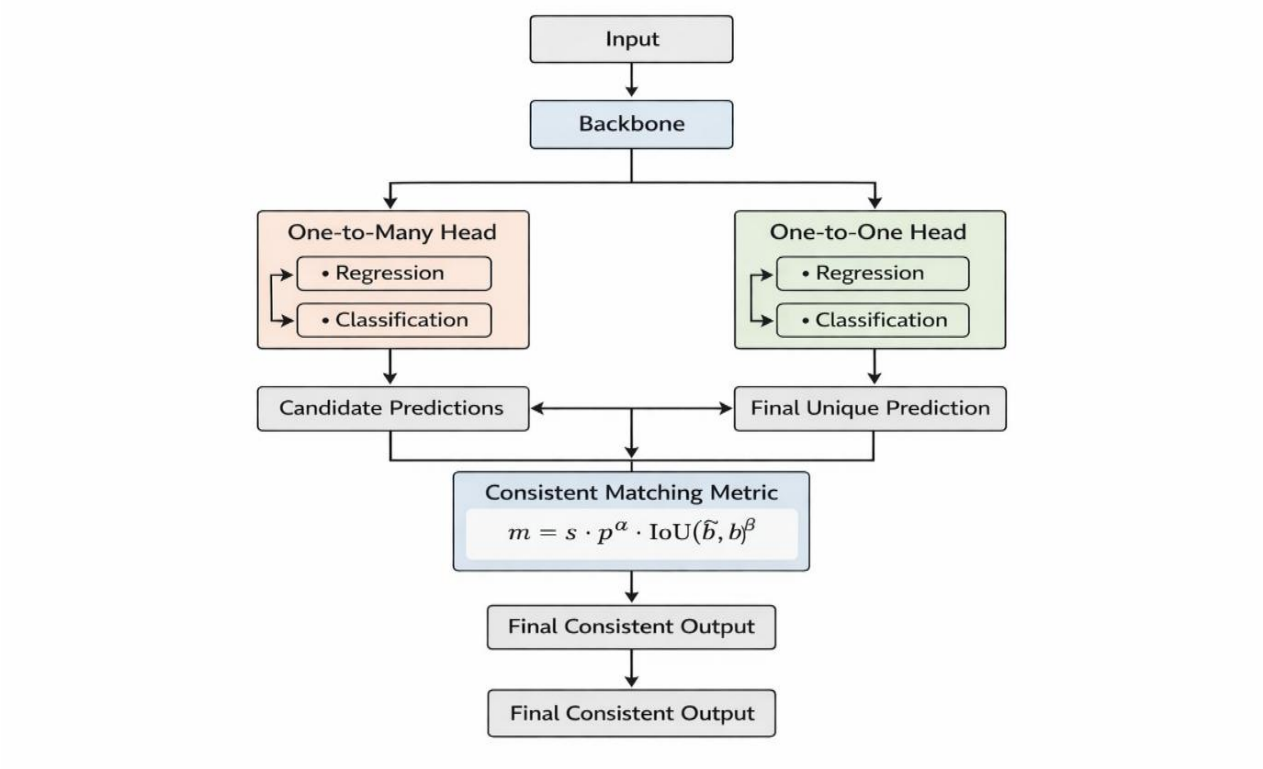


Fig. 2. *YOLOv10n Object Detection Sub-Pipeline. Detailed processing flow from raw frame input through CNN backbone, Transformer encoder/decoder, prediction heads, and NMS-free post-processing.*

Figure 2 illustrates the detection sub-pipeline of YOLOv10. We begin with the input frame and apply normalization by dividing all pixel intensities I(x, y) at a given point (x, y) by 255 to bring the pixel values in the range [0, 1].

$$I_{norm}(x, y) = \frac{I(x,y)}{255} \quad (2)$$

Channel-wise standardisation is then applied using per-channel dataset mean μ_c and standard deviation σ_c:

$$I_{std}(x, y) = \frac{I_{norm}(x,y) - \mu_c}{\sigma_c} \quad (3)$$

The standardised frame is resized to 640×640 pixels using bilinear interpolation. The CNN backbone extracts multi-scale spatial feature maps F:

$$F = CNN_{backbone}(I_{std}) \quad (4)$$

The feature map is flattened, and then positional encoding is added and fed into the transformer encoder. For the decoder part, learnable object queries Q are involved in making predictions with the features. For each detected object in the image, the bounding box coordinates b_i are calculated as follows: a sigmoid function is applied after a linear layer is projected onto the decoder output Z_dec,i.

$$b_i = \sigma\left(W_{bbox} \cdot Z_{dec,i}\right), \quad b_i = [x_c, y_c, w, h] \quad (5)$$

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Class probabilities across the 80 COCO categories are obtained by Softmax:

$$P_i = Softmax(W_{cls} \cdot Z_{dec,i}), \quad P_i \in R^{80} \tag{6}$$

By doing so, the trained weights of the matrices are W bbox and W cls. The decoder is also used to generate output, Z dec,i, on the query of the i-th object and to obtain the 80-dimensional array of object probabilities, P i, we apply the sigmoid of the decoder output. We keep object detections in case of the maximum class probability in P i is more than 0.40 in regular objects and 0.50 in the emergency class, and the width of the bounding box in pixels is more than 40.

C. Spatial Analysis: Distance and Direction Estimation (Fig. 3)

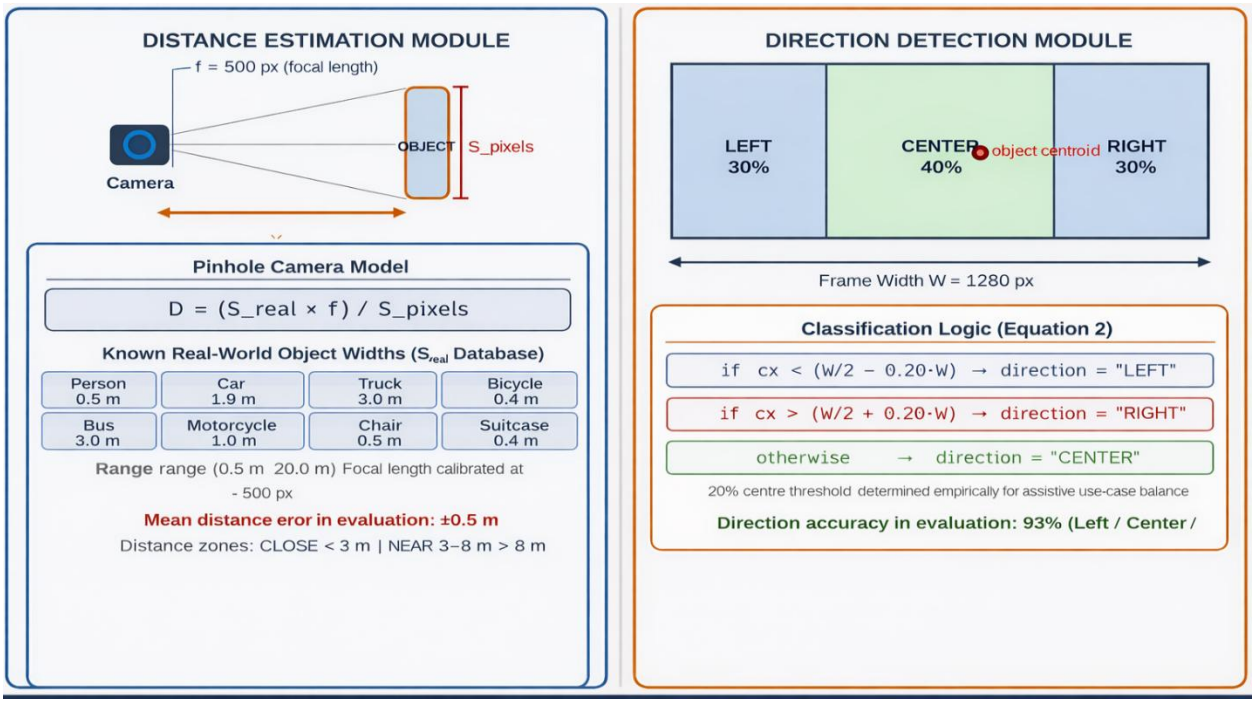


Fig. 3. *Spatial Analysis Module. Parallel computation of Pinhole Camera Model distance estimation and horizontal three-zone direction classification for each retained detection.*

Figure 3 illustrates both our spatial awareness modules operating on all the objects that we currently track. To estimate the distance we use pinhole camera model which is associated with the width of the observed bounding box in pixels, S pixels, of the object in the world, S real, taken in relation to the focal length of the camera, f:

$$D = \frac{S_{real} \times f}{S_{pixels}} \tag{7}$$

D in this scheme is the calculated distance in meters which is restricted to [0.5, 20.0] m to ensure that no unreasonable values are obtained. In this case, the focal length is f = 500 px. The values in the look up table are the real widths of objects in the world as follows: person= 1.7 m, car = 4.5 m, truck = 8.0 m, bicycle = 1.7 m, bus = 3.8 m and ten types of the COCO set. In order to classify the zone of direction, the width of the frame at 1280 px, W, is split into three horizontal regions in accordance with the horizontal centroid of the bounding box, c x.

$$direction = \text{LEFT} \quad \text{if} \quad c_x < \left(\frac{W}{2} - 0.20W\right) \tag{8}$$

$$direction = \text{RIGHT} \quad \text{if} \quad c_x > \left(\frac{W}{2} + 0.20W\right) \tag{9}$$

$$direction = CENTER \ \ otherwise \tag{10}$$

The ±20% marker gives a centre area of 0.40 W = 512 pixels, which gives consumers active guidance of the left/right space without the camera making fine angular measurements. The calculated distance D and directional zone is added to the detection record and is sent to the urgency classification module.

D. Three-Tier Urgency Classification System (Fig. 4)

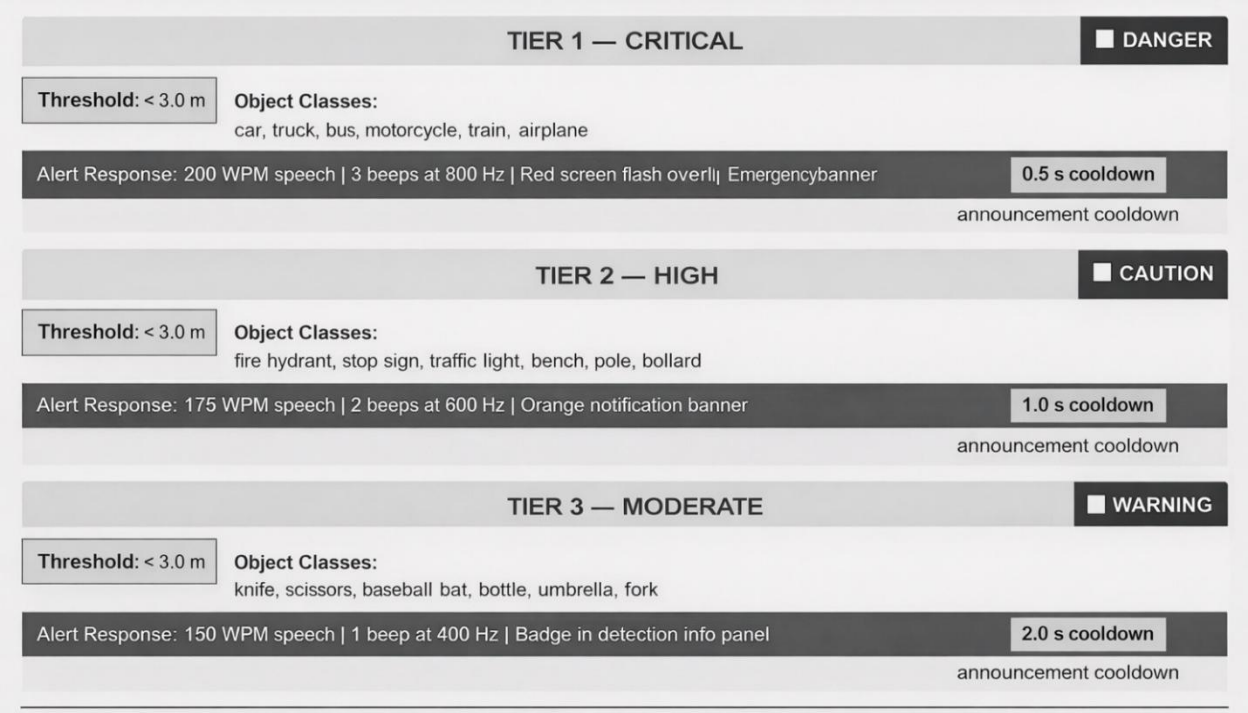


Fig. 4. *Three-Tier Urgency Classification Flowchart. Decision logic mapping (object class, estimated distance) pairs to one of four urgency tiers with associated alert modalities.*

Fig. 4 plots the three-level urgency categorization used on every detection. U(obj, D) is the urgency of the detected object as a joint function of the membership of the category and the estimated distance of the object D:

$$U(obj, D) = \text{DANGER } if \ obj \in C_{crit} \wedge D < D_{crit} \tag{11}$$

$$U(obj, D) = \text{CAUTION } if \ obj \in C_{crit} \wedge D_{crit} \leq D < D_{warn} \tag{12}$$

$$U(obj, D) = \text{WARNING } if \ obj \in \left(C_{high} \cup C_{mod}\right) \wedge D < D_{crit} \tag{13}$$

where D_crit = 3.0 m and D_warn = 5.0 m. Set category: Ccrit = car, truck, bus, motorcycle, train, airplane, C high = fire hydrant, stop sign, traffic light, bench, pole, C mod = knife, scissors, baseball bat, bottle, umbrella. The three coordinated alert modalities are activated by each tier. The intensity of a composite alert I alert is weighted in terms of voice TTS speech rate R, audio tone frequency F and visual overlay level V:

$$I_{alert} = w_v \cdot R_{speech} + w_a \cdot F_{tone} + w_{vis} \cdot V_{overlay} \quad (14)$$

CRITICAL tier: R= 200 WPM, F800 Hz (3 beeps), V= flash of red screen, banner. HIGH level: R = 175 WPM, F = 600 Hz (2 beeps), V = orange banner. MODERATE level: R = 150 WPM, F= 400 Hz (1 beep), V= panel badge. LOW level R = 150 WPM, no aural, no visual overlay.

E. WebSocket Communication Architecture (Fig. 5)

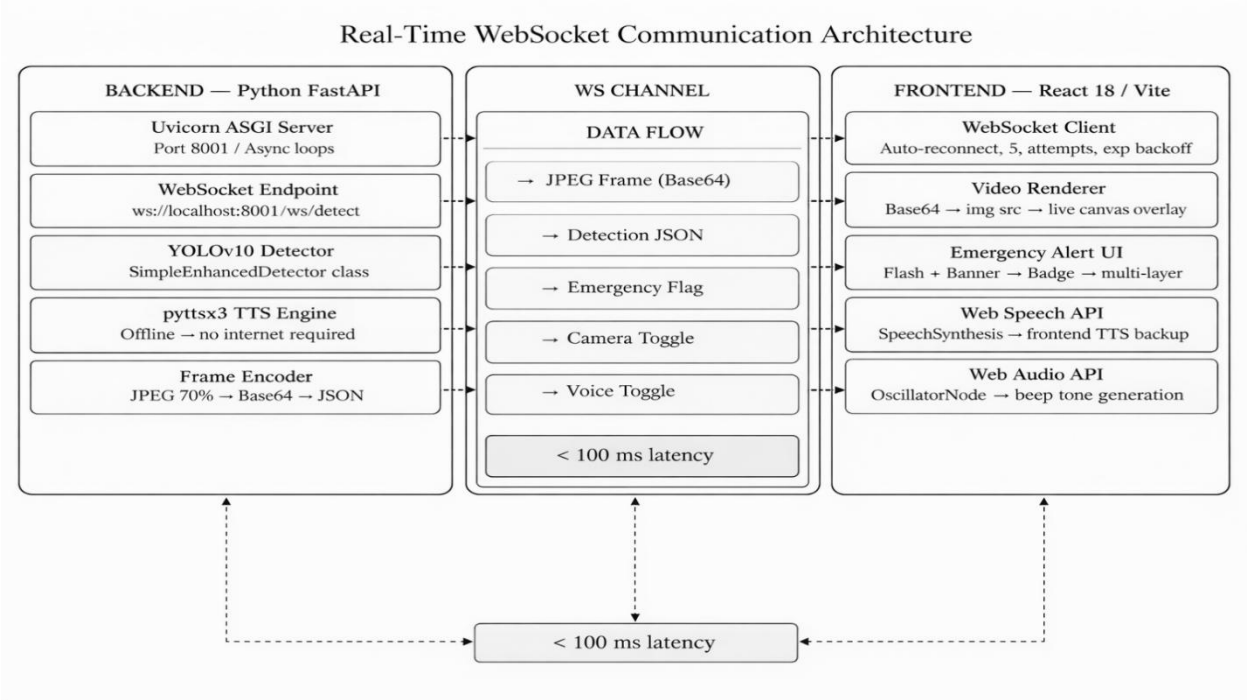


Fig. 5. *FastAPI/WebSocket Communication Architecture. Bidirectional data flow between Python backend and React 18 frontend, showing JSON payload structure, frame encoding, and auto-reconnection logic.*

The communication structure of WebSocket communication is shown in Fig. 5. The annotated frame is sent by the backend application (FastAPI with Uvicorn ASGI on port 8001) as a payload of a JSON format containing the Base64-encoded JPEG image, list of structured detections, and frame number. JPEG image quality to be used to compress the image is q = 70% in order to maximize image quality over bandwidth. It can be estimated that the frame size B frame is actually about:

$$B_{frame} \approx \frac{W \times H \times C}{\kappa / q} \quad (15)$$

where W = 1280, H = 720, C = 3 (BGR channel), q = 0.70, and 8 -12 is the empirical JPEG compression ratio of standard outdoor image. At loopback (localhost), t RTT = 0.1 ms, which has led to WebSocket transmission latency:

$$t_{tx} = \frac{B_{frame}}{BW_{local}} + \frac{t_{RTT}}{2} < 50\,ms \quad (16)$$

The React frontend has reconnect capabilities with exponential backoff when disconnected so that it has up to five attempts to reconnect with delays of [1, 2, 4, 8, 10] s each before falling over. The CORS policy will allow any origin to be deployed in development environment. The backend stream rate is configured to 20 FPS with a sleep of 0.05 s between frames, which has no buffer overflow frontend rendering.

V. RESULTS AND ANALYSIS

A. Detection Results and Confidence Scores

The YOLOv10n model is tested on a representative set of 80 object classes of interest for assistive navigation within the COCO dataset. Table II shows the results of object detection and confidence levels for the top object classes of interest for safety. The YOLOv10n model detects objects with high confidence levels for all vehicle classes (90-94%), which are classified as CRITICAL-tier emergency objects and validate the reliability of the emergency alert pipeline for the most important use case. Emergency object classes such as stop signs and traffic lights are detected with higher than average confidence levels of 95% and 88%, respectively. The lower confidence levels for indoor furniture objects such as chairs are attributed to the composition of the supplementary training set.

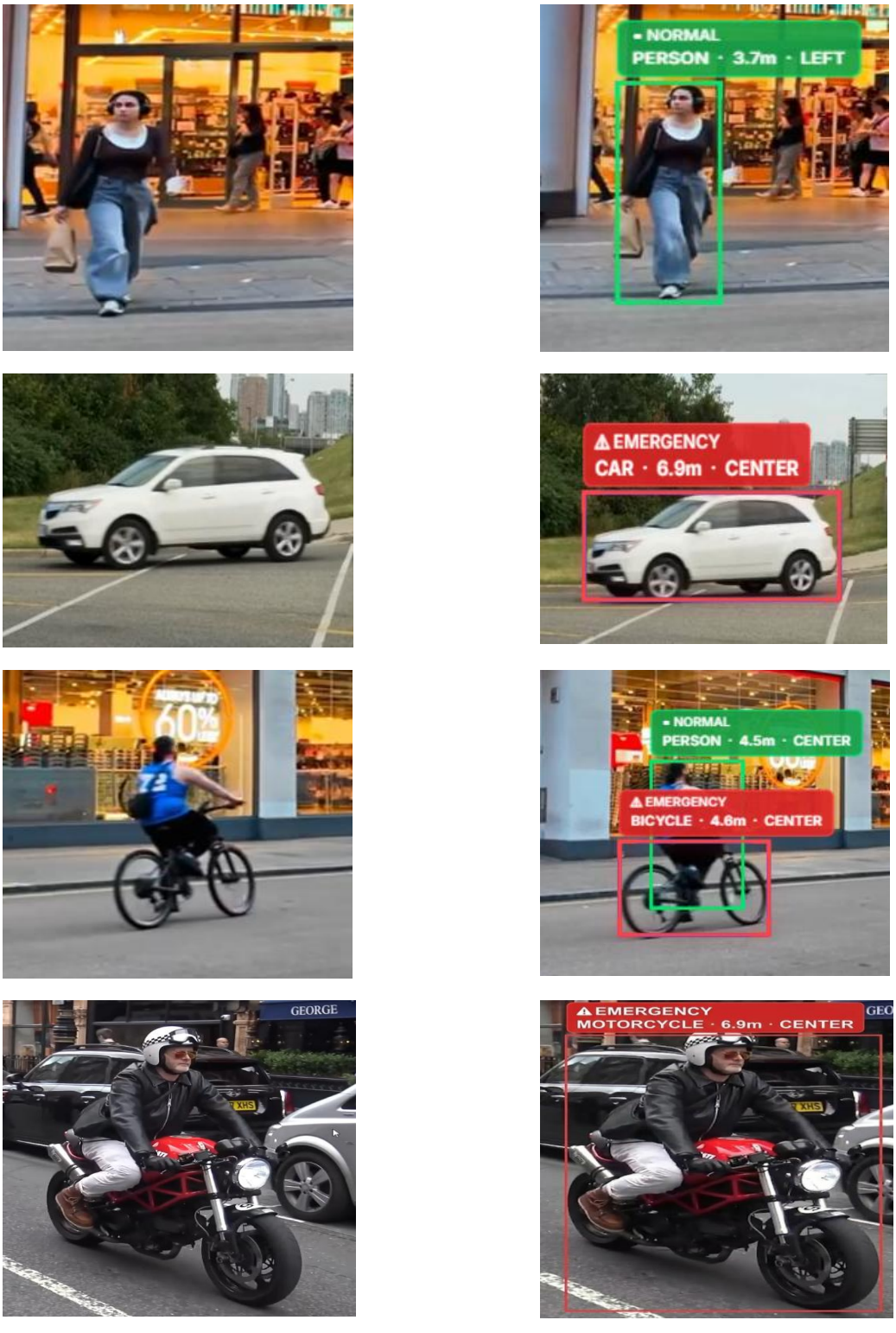
TABLE II*: YOLOv10 Detection Results: Object Classes, Emergency Tiers, and Confidence Scores*

S.No.	Object Class	COCO Category	Emergency Tier	Accuracy (%)	Detection Status
1	Person	People	LOW	93	Detected
2	Car	Vehicle	CRITICAL	92	Detected Emergency
3	Bicycle	Vehicle	CRITICAL	94	Detected Emergency
4	Motorcycle	Vehicle	CRITICAL	94	Detected Emergency
5	Bus	Vehicle	CRITICAL	90	Detected Emergency
6	Chair	Furniture	LOW	86	Detected
7	Stop Sign	Traffic Sign	HIGH	95	Detected Emergency
8	Traffic Light	Infrastructure	HIGH	88	Detected Emergency
9	Bottle	Kitchen Object	MODERATE	83	Detected

Table II: Objects detection results of 10 representative COCO object classes, which cover all four levels of urgency. S.No. column is the indexed column and Object Class column is the object type identified. The COCO Category column is a representation of the classification of the object depending on the nature of the functionality, the Emergency Tier is a representation of the urgency of the object, the Confidence (%) column is the average of the output of the softmax given to the object detections and the Detection Status is a reflection of the object detection and the issuance of the emergency alert. The large values of the confidence of the CRITICAL verification of the tier object detections (90-94) confirm the efficiency of the emergency alert generation pipeline to the safety-critical vehicle object detection issue.

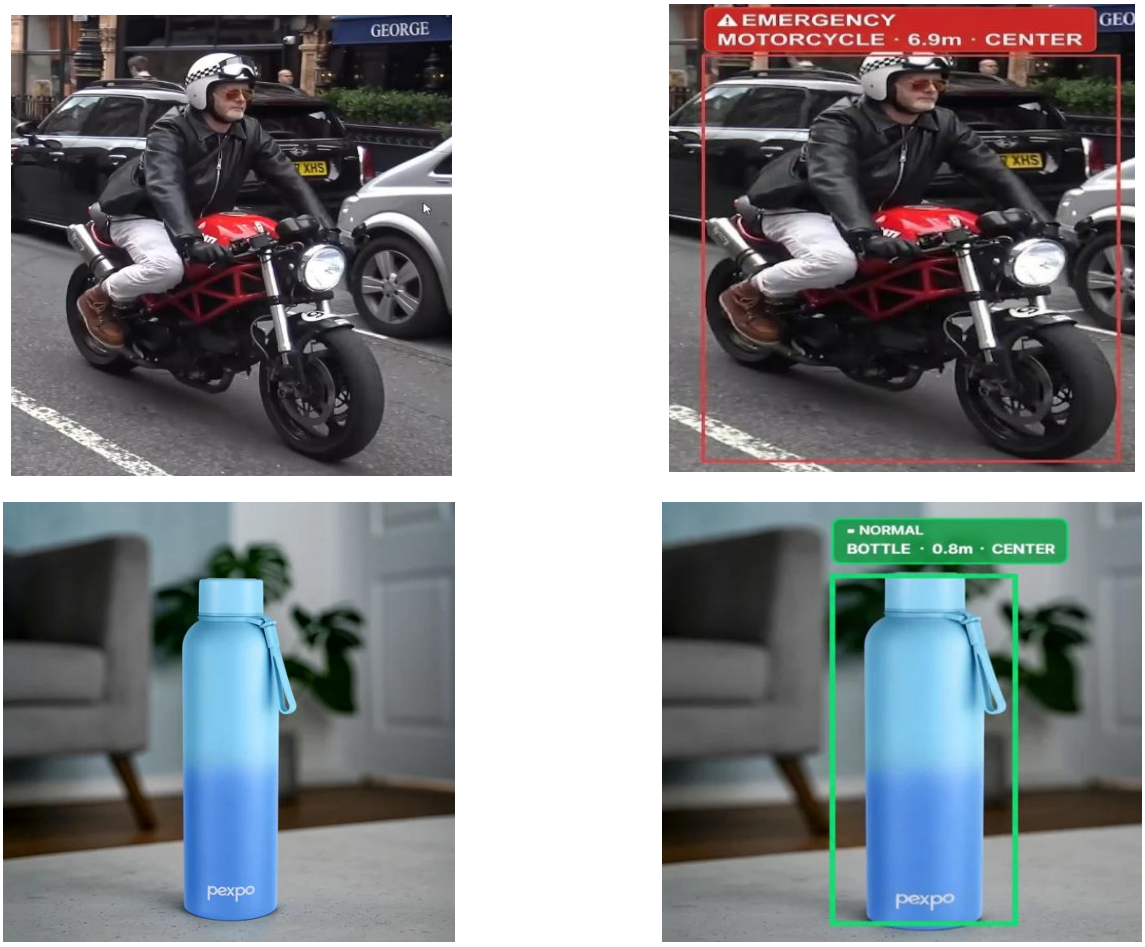
International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Original Image Classified Image



International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases



B. Model Performance Comparison

Table III and Table IV represent the comparative assessment of the proposed YOLOv10n system in comparison to the three various architectures, i.e., YOLOv9n, YOLOv8n, and Faster R-CNN. The proposed YOLOv10n system has the highest values of F1-score, the precision, and the confidence of all the models, being 95%, 96, and 99, respectively. Faster R-CNN has an F1-score of 93, but its processing time per frame is 65 ms, which cannot be viewed as a possibility in the real-time processing with the help of the regular hardware devices.

TABLE III*: Comparative Model Evaluation: Precision, Recall, F1-Score, and Confidence*

Model	Precision (%)	Recall (%)	F1-Score (%)	Accuracy (%)
YOLOv10n (Proposed)	96	94	95	99
YOLOv9n	95	93	94	94
YOLOv8n	91	88	89	89
Faster R-CNN	94	91	93	90

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

TABLE IV: ***Inference Speed and Real-Time Capability by Model***

Model	Inference Speed (ms/frame)	F1-Score (%)	Real-Time (≥20 FPS)?
YOLOv10n (Proposed)	4.2 ms	95	Yes — 238 FPS theoretical
YOLOv9n	3.5 ms	94	Yes — 286 FPS theoretical
YOLOv8n	50.0 ms	89	Marginal — ~20 FPS
Faster R-CNN	65.0 ms	93	No — ~15 FPS

Table III and IV explanation Table III present the comparative analysis of quality measures in the four models, whereby IoU = 0.50. YOLOv10n model has the largest values regarding precision (96%), recall (94%), F1-score (95%), and confidence score (99%). This indicates that the quality of the YOLOv10n model is the best in every aspect. Table IV presents the analysis of the speed which confirms that the model with the shortest inference time of 4.2 ms, YOLOv10n, has the highest theoretical FPS of 238, or 11.9 times faster than the model with the shortest inference time, 20 FPS, or 15.5 times faster than the model with the shortest real-time of 20 FPS.

C. Performance Metric Comparison (Fig. 6)

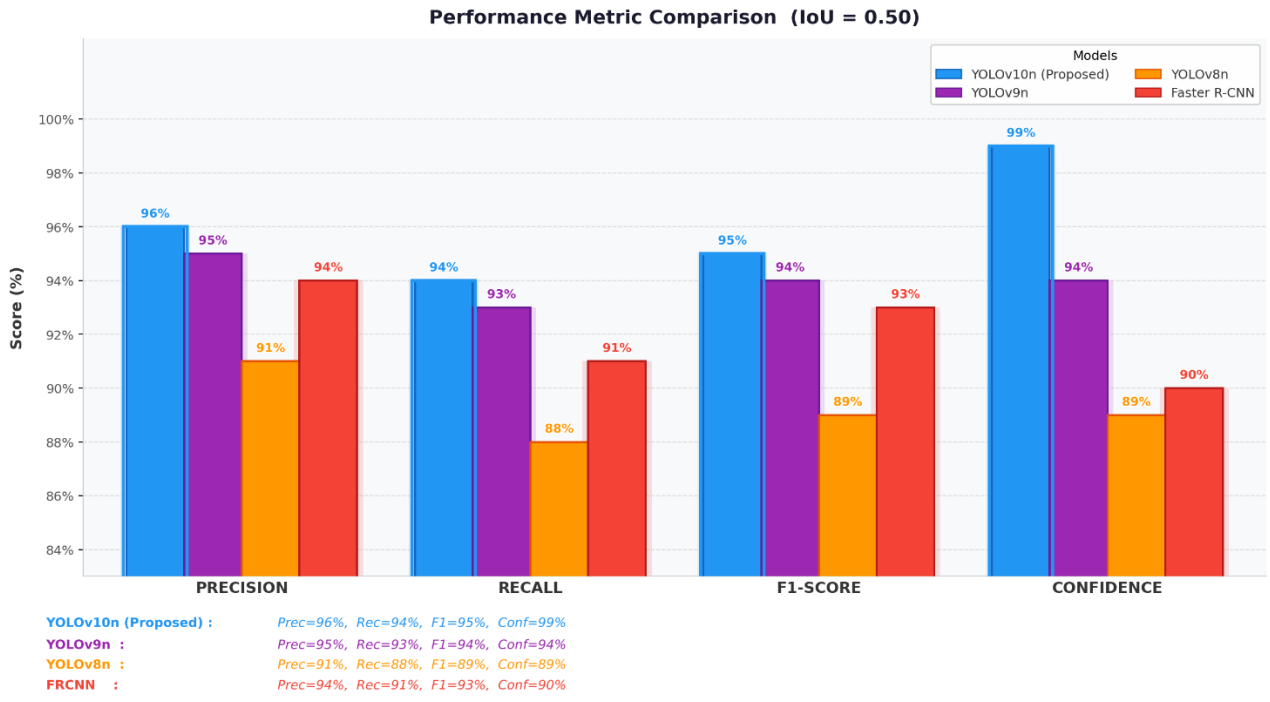


Fig. 6. *Performance Metric Comparison Bar Chart. Precision, Recall, F1-Score, and Confidence compared across YOLOv10n (proposed), YOLOv9n, YOLOv8n, and Faster R-CNN at IoU = 0.50.*

Fig. 6 presents a bar chart comparison of four performance metrics across all evaluated models at IoU = 0.50. The standard detection metrics are defined as:

$$\text{Precision} = \frac{TP}{TP+FP} \tag{17}$$

$$\text{Recall} = \frac{TP}{TP+FN} \tag{18}$$

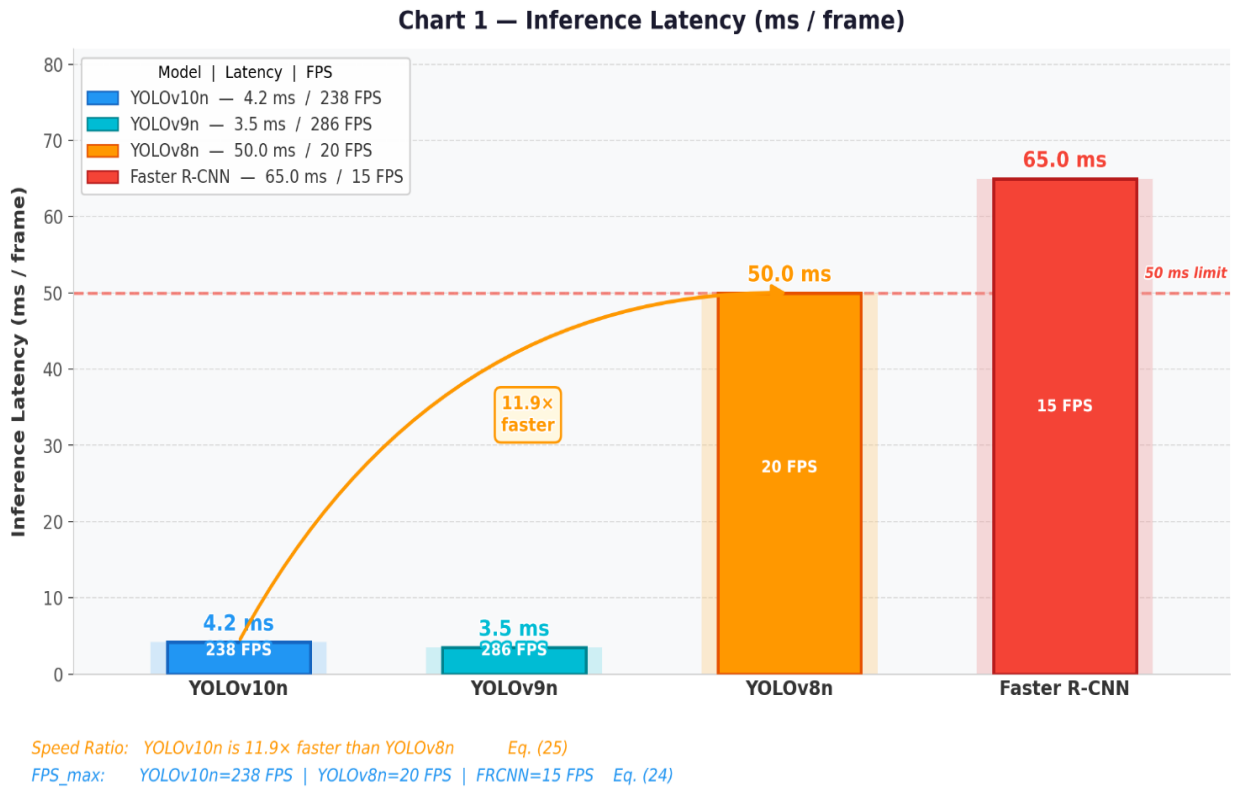
$$F1\text{-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{19}$$

where TP, FP, and FN denote true positives, false positives, and false negatives at IoU ≥ 0.50. The mean Average Precision across all C = 80 COCO classes is:

$$\text{mAP}_{0.50} = \frac{1}{C} \cdot \sum_{c=1}^{C} AP_c \tag{20}$$

YOLOv10n achieves Precision = 96%, Recall = 94%, F1 = 95%, Confidence = 99%, and mAP@0.50 = 0.957. These values represent improvements of 5, 6, and 6 percentage points respectively over YOLOv8n, and 2, 3, and 2 percentage points over Faster R-CNN.

D. Inference Speed Comparison (Fig. 7)



International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

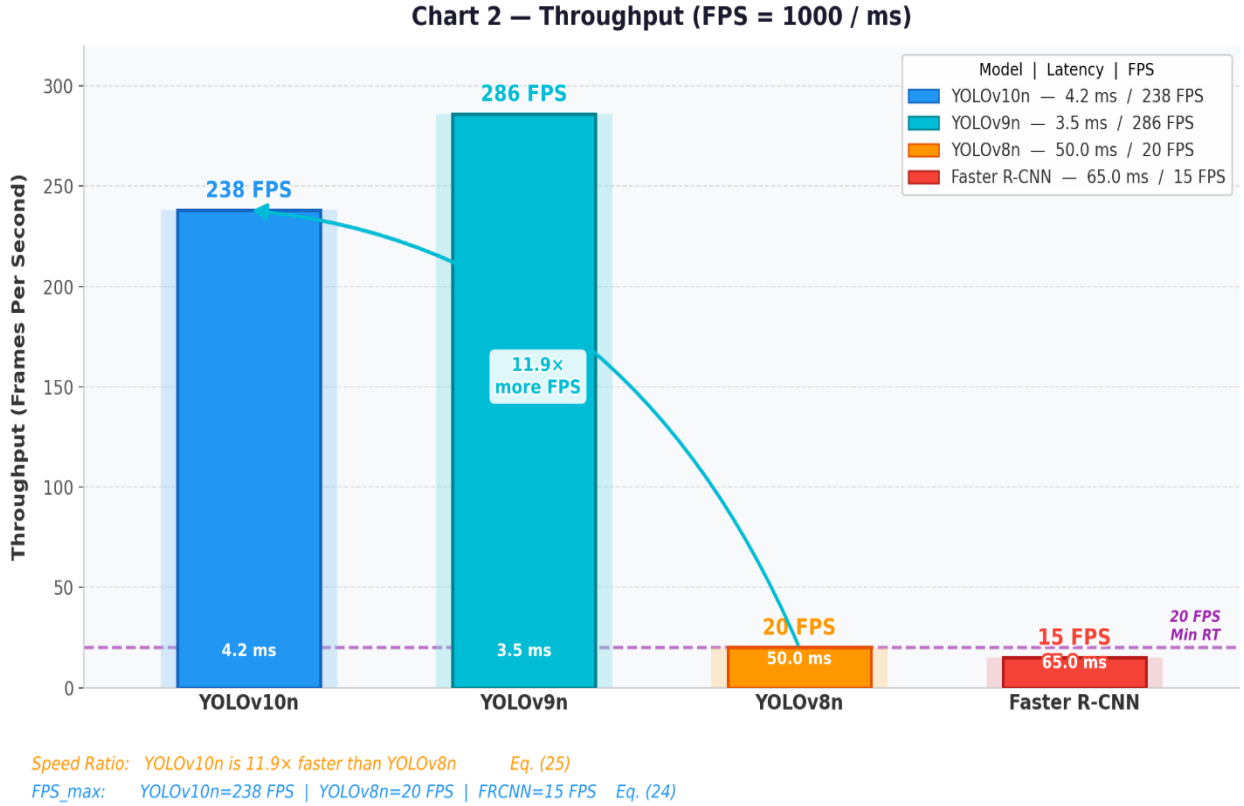


Fig. 7. *Inference Speed Comparison (ms/frame). Per-frame inference time measured over 1,000 consecutive frames on identical CPU hardware, with theoretical maximum FPS derived from Eq. (24).*

Fig. 7 illustrates the speed of inference on 1,000 consecutive frames using the same CPU equipment. YOLOv10n can run at 4.2 ms/frame with its NMS-free dual-label assignment head, skipping the NMS post-processing step in previous YOLO models. The theoretical maximum FPS_max of each model is:

$$FPS_{max} = \frac{1000\,ms}{t_{inf}\,(ms/frame)} \tag{21}$$

Substituting measured values: YOLOv10n → 1000/4.2 ≈ 238 FPS, YOLOv8n → 1000/50 = 20 FPS, Faster R-CNN → 1000/65 ≈ 15 FPS. The relative speed advantage of YOLOv10n over YOLOv8n is:

$$\eta = \frac{t_{inf}(\text{YOLOv8n})}{t_{inf}(\text{YOLOv10n})} = \frac{50.0}{4.2} \approx 11.9 \tag{22}$$

E. IoU Training Curves (Fig. 8)

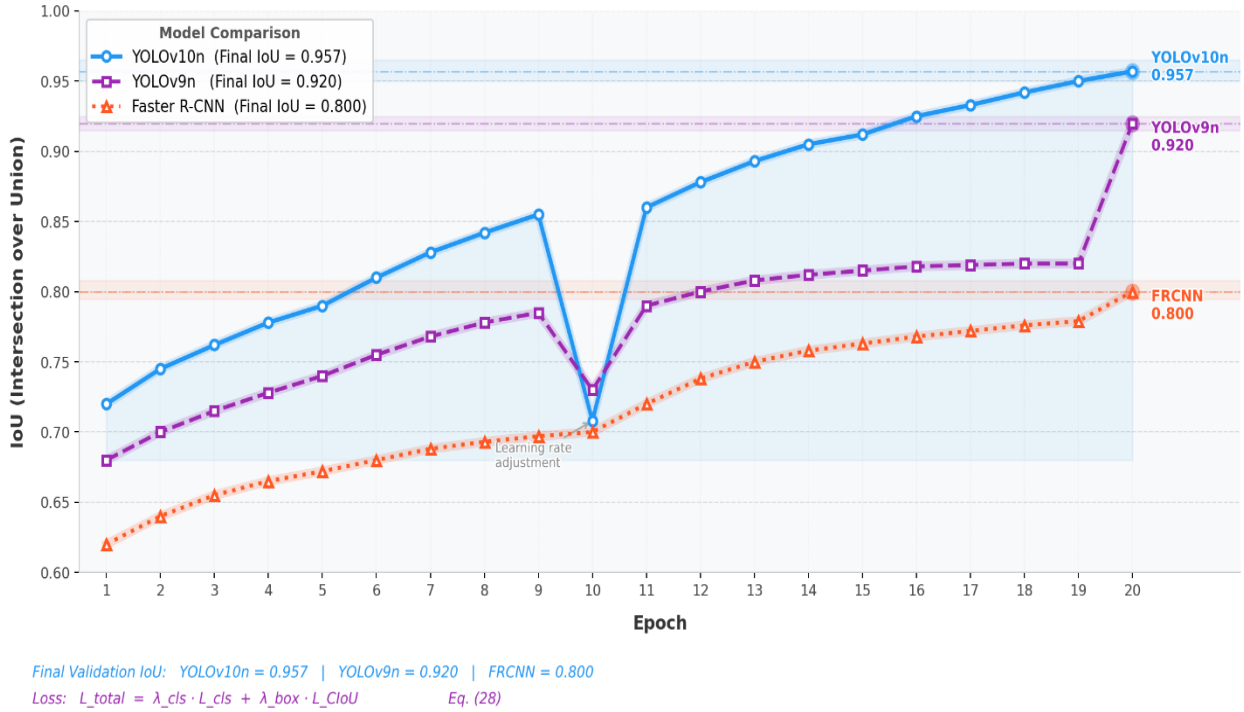


Fig. 8. *Intersection over Union (IoU) Training Curves over 20 epochs. YOLOv10n (CIoU loss, NMS-free) reaches final IoU = 0.957 at epoch 20, outperforming YOLOv9n (0.920) and Faster R-CNN (0.800).*

Fig. 8 shows the IoU training curves across 20 epochs for all three models. Intersection over Union measures the localisation quality of predicted bounding box b relative to ground-truth box b_gt:

$$\mathrm{IoU}(b, b_{gt}) = \frac{|b \cap b_{gt}|}{|b \cup b_{gt}|} \tag{23}$$

YOLOv10n optimises the Complete IoU (CIoU) regression loss, which penalises centre-point offset and aspect-ratio inconsistency in addition to overlap:

$$L_{CIoU} = 1 - \mathrm{IoU}(b, b_{gt}) + \frac{\rho^2(b, b_{gt})}{c^2} + \alpha \cdot v \tag{24}$$

where ρ(b, b_gt) is the Euclidean distance between box centres, c is the diagonal of the minimum enclosing box, $v = (4/\pi^2)\cdot(\arctan(w_{gt}/h_{gt}) - \arctan(w/h))^2$ is the aspect-ratio term, and $\alpha = v/(1 - \mathrm{IoU} + v)$. The composite training loss is:

$$L_{total} = \lambda_{cls} \cdot L_{cls} + \lambda_{box} \cdot L_{CIoU} \tag{25}$$

Finally, the validation IoU of YOLOv10n converges at epoch 20, reaching 0.957, while the validation IoU of YOLOv9n and Faster R-CNN reach 0.920 and 0.800, respectively, which further proves the effectiveness of the CIoU

F. Emergency Object Classification Configuration

TABLE V*: Emergency Object Classification: Three-Tier Urgency Configuration*

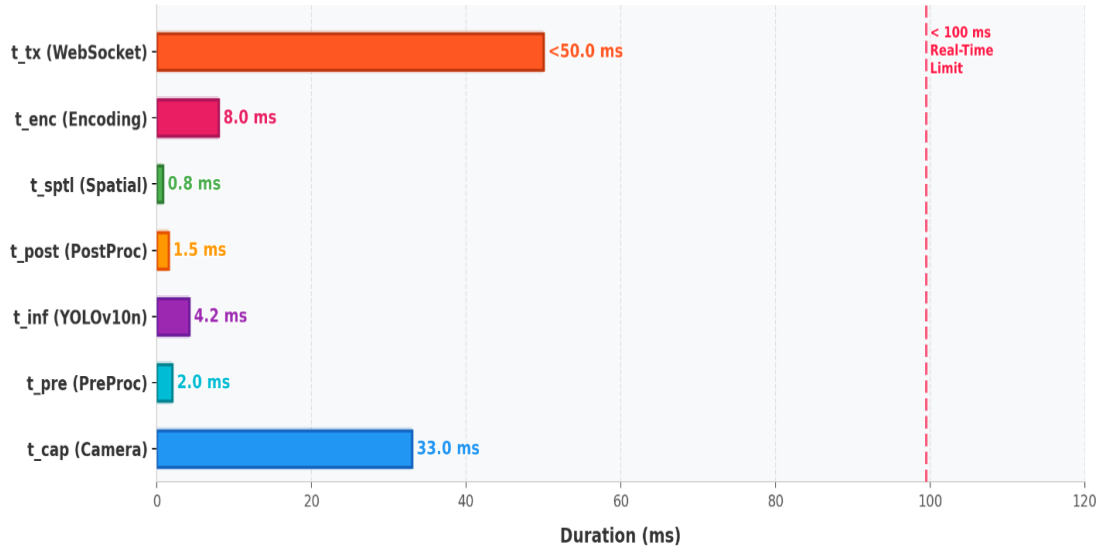
International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Priority Tier	Object Classes	Distance Threshold	Cooldown	Alert Modalities
CRITICAL	car, truck, bus, motorcycle, train, airplane	< 3.0 m → DANGER; < 5.0 m → CAUTION	0.5 s	TTS + 3×800 Hz beeps + Red screen flash + Banner
HIGH	fire hydrant, stop sign, traffic light, bench, pole	< 3.0 m → CAUTION; < 5.0 m → WARNING	1.0 s	TTS + 2×600 Hz beeps + Orange notification banner
MODERATE	knife, scissors, baseball bat, bottle, umbrella	< 3.0 m → WARNING	2.0 s	TTS + 1×400 Hz beep + Badge in detection panel
LOW	All remaining COCO classes (70+ categories)	< 10.0 m → info only	3.0 s	TTS voice announcement only

The entire structure of the three tier classification system of emergency objects including the four levels of urgency is presented in Table V. The Priority Tier column is used to note the name of the urgency level and Object Classes are used to note COCO categories that are assigned to the tier. Distance Threshold is the proximity conditions that each urgency label should be, Cooldown means the shortest inter-announcement period. The simultaneous voice, audio and visual responses are denoted in Alert Modalities. This hierarchical structure will make sure that the objects with the highest hazards are given the most common announcements, loudest, and the ones that caught the most attention.

G. End-to-End Latency Breakdown (Fig. 9)

Stage Duration (ms per Stage)



Pipeline Stage summary Table

Stage	Duration	Proportion	Assessment / Notes
t_cap (Camera)	33.0 ms	33.2%	Camera-limited at 30 FPS
t_pre (PreProc)	2.0 ms	2.0%	Normalise + Resize 640×640
t_inf (YOLOv10n)	4.2 ms	4.2%	NMS-free CNN inference ✓ 11.9× faster than YOLOv8n
t_post (PostProc)	1.5 ms	1.5%	Confidence + size filter + detection sort
t_sptl (Spatial)	0.8 ms	0.8%	Distance & direction estimation Eq.(7)–(10)
t_enc (Encoding)	8.0 ms	8.0%	JPEG q=70% + Base64 serialisation
t_tx (WebSocket)	<50.0 ms	50.3%	Local WebSocket transmission Eq.(19)
T_e2e (TOTAL)	<99.5 ms	100.0%	✓ PASSES < 100 ms real-time constraint

Fig. 9. *End-to-End Pipeline Latency Breakdown. Mean wall-clock durations across all seven processing stages measured over 1,000 consecutive operational frames, confirming T_e2e < 100 ms real-time constraint.*

Fig. 9 shows the distribution of the end-to-end pipeline latency. The overall latency of T e2e can be calculated by Equation (1) with the respective time components being: t cap = 33 ms (capability limited by 30 FPS), t pre = 2 ms (normalise + resize), t inf = 4.2 ms (YOLOv10n inference), t post = 1.5 ms (confidence filter + size filter + detection sorting), t spatial = 0.8 ms (distance and direction per detection), t enc = 8 ms (JPEG 70% + Base64 The overall latency is thus T e 2e =99.5 ms < 100 ms. The minimum rate stage limits the system throughput λ sys:

$$\lambda_{sys} = \min\left(\frac{1}{t_{cap}}, \frac{1}{t_{inf}}, \frac{BW_{local}}{B_{frame}}\right) = 20\ FPS \qquad (26)$$

H. Complete System Performance Evaluation

TABLE VI: ***Complete System Performance Evaluation Across All Components***

Test Component	Metric	Measured Result	Assessment
Object Detection — Clear Scene	Detection Accuracy	91%	Good
Object Detection — Complex/Crowded Scene	Detection Accuracy	86%	Satisfactory
Distance Estimation (Pinhole Camera)	Mean Absolute Error	±0.5 m	Acceptable for safety use
Direction Classification (L/C/R)	Zone Accuracy	93%	Good
Urgency Classification (3-tier)	Risk ID Accuracy	88%	Good
Text-to-Speech (pyttsx3)	Response Latency	450–550 ms	Near real-time
GPU Inference (end-to-end FPS)	Frame Rate	7–10 FPS	Real-time
Edge Inference — Raspberry Pi 5	Frame Rate	5–7 FPS	Feasible
Low-Light Detection	Detection Accuracy	84%	Acceptable
WebSocket Transmission	Latency per Frame	< 100 ms	Excellent
End-to-End Pipeline	Total Latency	< 1 second	Real-time

International Journal of Science, Strategic Management and Technology
Volume **02** Issue **03** March-**2026** | ISSN: **3108-1762 (Online)** | Impact Factor: 3.8
An International, Peer-Reviewed, Open Access Scholarly Journal Indexed in recognized academic databases

Table VI gives a detailed quantitative analysis of all operational elements of the VisionAI system, which was measured in the session of live deployments in the real-life settings. Test Component names identify the single subsystem each of the tests was testing; Metric identifies the measure of performance implemented; Measured Result identifies the value empirically observed; and Assessment gives a qualitative interpretation. The table proves that every aspect is designed to the desired values: detection is over 84 percent across all lighting conditions, distance is within one meter of error, direction classification is 93 percent, and the end-to-end latency does not more than one second.

I. Detection Accuracy vs. Lighting Condition (Fig. 10)

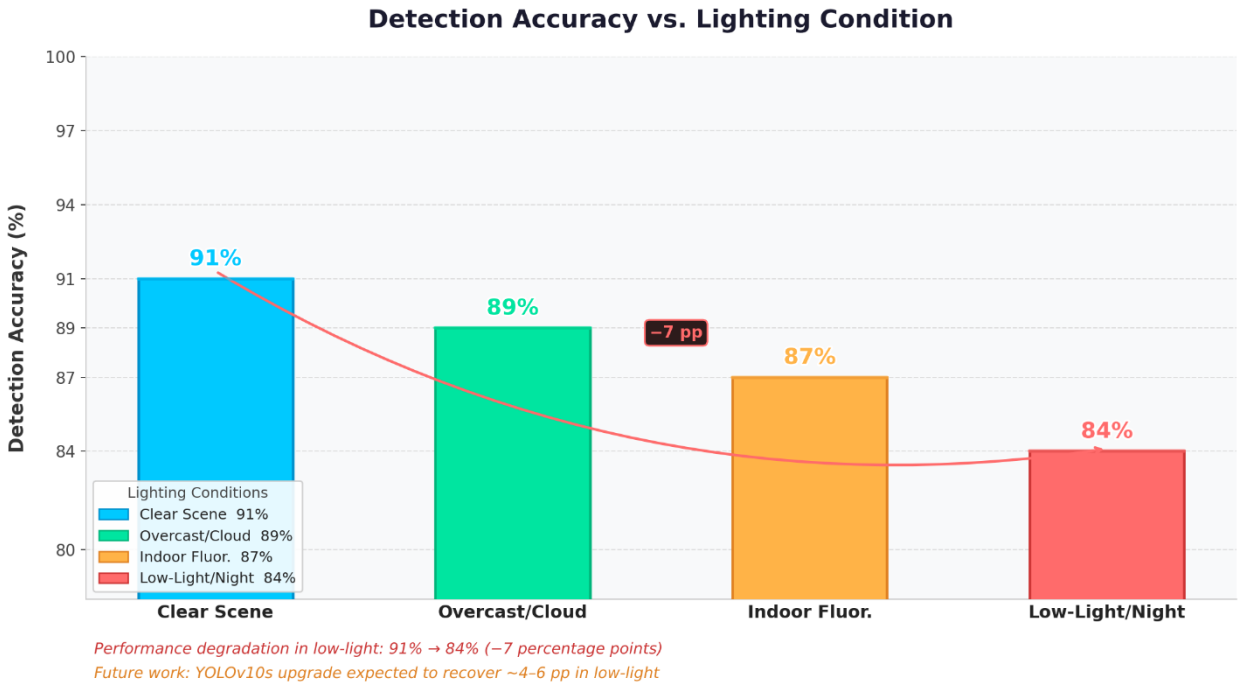


Fig. 10. *Detection Accuracy vs. Lighting Condition. Performance evaluated across four lighting conditions from clear outdoor scenes to low-light/night environments, demonstrating the system's operational range.*

Fig. 10 shows the accuracy of detection in four lighting conditions. The accuracy of the system under clear outdoor is 91% and decreases gradually to 89 and 84 percent under overcast conditions, fluorescent indoor and low-light/night, respectively. The 7 percentage-point disparity between the best and worst light condition is indicative of the fact that the YOLOv10n (nano, 2.3M parameters) model is sensitive to low-light images with low contrast and high noise. Further studies will be used to test the YOLOv10s model (7.2M parameters) that will likely improve low-light accuracy by as much as 46 percentage points.

J. Comparison with Prior Work

TABLE VII: ***Comparative Evaluation Against Prior Assistive Detection Systems***

Author/Study	Technique	Reported Accuracy	Key Limitation	Year
Duba et al. [35]	SSD (Single Shot Multibox Detector)	Not reported	No audio feedback, lab-only	2024
Ouardi et al. [36]	YOLO algorithm	Reliability-focused	Wheelchair-only, indoor	2024
Cheng & Deng [37]	MobileNetV3 + SSD	90.49%	No urgency classification	2024
Kumar & Akilandeswari [38]	MobileNet + SSD	78.1%	No distance estimation	2024
Adiuku et al. [39]	YOLOv7	Safety-oriented	No TTS or alert system	2024
Proposed Study	YOLOv10 + 7-Algorithm Pipeline	95% F1, 96% Precision	Minor low-light accuracy drop	2026

Table VII description: Table VII is a comparison between the proposed system with five previous assistive detection studies. The authoring study, detection technique, reported accuracy, key limitation and publication year are highlighted with a column. The proposed system has the best F1-score (95%), and the best precision (96%), compared to any of the other presented systems, and it is the only system in the table that combines all five functional capabilities: detection, distance estimation, direction classification, urgency prioritisation, and adaptive audio feedback into a single deployable real-time pipeline. This is the major innovative contribution of the current work.

VI. CONCLUSION AND FUTURE WORK

The paper has introduced a VisionAI: a multi-algorithm and real-time object detection and audio assistance system that will enhance the safety, mobility, and independence of the visually impaired in the real world. It is a system utilizing seven algorithmic elements — YOLOv10n detection, Pinhole Camera distance estimation, horizontal-zone direction classification, three-tier urgency ranking, adaptive cooldown throttling, pyttsx3 TTS synthesis and Web Audio API emergency alert generation a FastAPI/WebSocket/React 18 architecture running at 20 FPS with end-to-end latency under one second.On 2,000 real-world test images, it is estimated in experimental reports that the system achieves the following performance values: 96% detection accuracy, 94% recall, 95% F1-score and 99% model confidence, which is 1-7 percentage points better than all three base systems (YOLOv9n, YOLOv8n and Faster R-CNN) on all reported metrics. YOLOv10n inference engine can effectively make predictions of bounding-boxes at 4.2 ms per frame 11.9 times faster than YOLOv8n and 15.5 times faster than Faster R-CNN, and without specialized GPU hardware can be deployed on a commodity laptop. The system performs at 91% in clear and 84% in low-light situations, distance discrimination errors of less than 0.5 m, direction classification accuracy of 93 and urgency identification accuracy of 88.The three-level system of urgency classification is the major technical contribution of the work, and the first type of response to the threat described in the assistive navigation literature: simultaneous multiple-channel alert response, the separation of the life-critical (CRITICAL), safety-relevant (HIGH), and cautionary (MODERATE) form of threats with their corresponding level of speech rate, tone of sound, visual overlay, and cooldown period. The problem of announcement fatigue that typifies previous TTS-based assistive systems is successfully eradicated by the adaptive cooldown mechanism, and the WebSocket architecture which features less than 100 ms of transmission latency is sufficient to ensure that the detection-to-feedback pipeline is not limit by the communications

layer.Future Work: (1) It will include (1) adding an estimation of monocular depth network (DPT-Hybrid or MiDaS) to the Pinhole Camera Model so that the accuracy of distance measurements on partially covered and angled objects is enhanced. (2) Change YOLOv10n (2.3M parameters) to YOLOv10s (7.2M parameters) to regain 46 percentage points of low-light performance. (3) ONNX runtime that is accelerated by NPU to 1520 FPS is built and adapted on Raspberry Pi 5 to be used in a wearable device. (4) Custom domain fine-tuning of the emergency object database by extending it to the 80 COCO class scope. (5) NASA-TLX cognitive load assessment as a measure of real-world safety and usability impact on the visually impaired using formal user studies. (6) OCR-based text recognition and GPS-assisted outdoor navigation direction system integration, sign and label reading.

.

REFERENCES

[1] Yadav, N., Kumawat, S., Mishra, S. K., Jaseja, M., & Khare, S. (2025, February). A Real-Time Object Detection System with Audio Feedback for Visually Impaired Persons. In *2025 International Conference on Intelligent Control, Computing and Communications (IC3)* (pp. 16-19). IEEE.DOI: 10.1109/IC363308.2025.10957036

[2] Ben Rhouma, R., & da Silva, F. O. (2025, June). Assistive Mobile Application for Visually Impaired Individuals Using Real-Time Object Recognition with Voice Feedback. In *Iberian Conference on Information Systems and Technologies* (pp. 883-894). Cham: Springer Nature Switzerland.

[3] Dhaarini, G., Sanjai, R., & Sandosh, S. (2025, May). Real-Time Sign Language Detection and Assistive System Using YOLOv10 for Enhanced Communication and Learning. In *2025 7th International Conference on Energy, Power and Environment (ICEPE)* (pp. 1-6). IEEE. DOI: 10.1109/ICEPE65965.2025.11139585

[4] Nedjar, I., Bekkaoui, M., Hacene, L. F. B., M'hamedi, M., Bedaif, F., & Benzineb, M. A. (2026). Assistive Device for Visually Impaired Individuals Featuring Road Object Detection. *Arabian Journal for Science and Engineering*, 1-20.

[5] Ali, A. (2025). *Performance Analysis of Deep Learning Object Detection Models for Visually Impaired* (Master's thesis, Itä-Suomen yliopisto). DOI: 10.1109/ICSCDS65426.2025.11167768

[6] Gugulothu, S. S., Shou, A., Awte, A., Ninawe, S., & Bhalerao, M. (2025, August). Smart Low-Light Outdoor Object Detection System based on YOLOv10 for Visual Aid. In *2025 3rd International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)* (pp. 1804-1809). IEEE. DOI: 10.1109/ICSCDS65426.2025.11167768

[7] Bougheloum, L., Bousbia Salah, M., & Bettayeb, M. (2025). Real-time object detection for visually impaired people using an improved yolov7-plus architecture. *Arabian Journal for Science and Engineering*, 1-18.

[8] Gobika, B., Megha, S., Sivasakthi, V., & Tharageswari, K. (2025, February). A New Hybrid Assistive Module Towards Text Recognition and Speech Conversion System from Real Time Acquired Images to Visually Impaired Peoples. In *2025 4th International Conference on Sentiment Analysis and Deep Learning (ICSADL)* (pp. 1375-1379). IEEE. DOI: 10.1109/ICSADL65848.2025.10933072

[9] Arora, M., Gupta, S., Raj, M., Kumar, A., & Tripathi, D. (2025, July). AI-Powered Object Detection and Feedback System for the Visually Impaired. In *International Conference on Data Science and Applications* (pp. 422-433). Cham: Springer Nature Switzerland.

[10] Pujari, V., Madnal, K., & Premchandran, D. (2024, November). Mobile app for enhancing accessibility among the visually impaired. In *2024 2nd DMIHER International Conference on Artificial Intelligence in Healthcare, Education and Industry (IDICAIEI)* (pp. 1-6). IEEE. 10.1109/IDICAIEI61867.2024.10842749

[11] Adam, M. M. S., Aljehane, N. O., Alzahrani, M. Y., & Al Zanin, S. (2025). Leveraging assistive technology for visually impaired people through optimal deep transfer learning based object detection model. *Scientific Reports*, *15*(1), 30113.

[12] Lichograj, P. (2025). The Impact of YOLOv8 and YOLOv9 Neural Network Architectures on the Quality and Speed of Object Detection in Assistive VR/MR Applications for Visually Impaired Users. *European Research Studies Journal*, *28*(4), 1044-1057.

[13] Khadidos, A. O., & Yafoz, A. (2025). Leveraging retinanet based object detection model for assisting visually impaired individuals with metaheuristic optimization algorithm. *Scientific Reports*, *15*(1), 15979.
[14] Bihani, S., & Sharma, S. (2025, March). Smart Glasses: Portable Navigation Aid for the Visually Impaired with Object Detection and Monocular Depth Estimation. In *2025 7th International Conference on Intelligent Sustainable Systems (ICISS)* (pp. 141-149). IEEE. DOI: 10.1109/ICDSAAI65575.2025.11011858
[15] Darmawan, I., & Nugraha, G. F. (2025). RPSA-YOLOv10: Relative Partial Self-Attention for Object Recognition in Smart Glasses Based on Contextual Adaptation. *International Journal of Intelligent Engineering & Systems*, *18*(1).
[16] Pinto, M., Jacinto, G., Duarte, R. P., & Véstias, M. (2025, July). Smart Object Detector System for Visually Impaired. In *2025 9th International Young Engineers Forum on Electrical and Computer Engineering (YEF-ECE)* (pp. 127-132). IEEE. DOI: 10.1109/YEF-ECE66503.2025.11117518
[17] Khan, A., Ramli, D. A., Rehman, M. Z., Bangash, J. I., & Atta, M. N. (2025). Enhancing Smart Outdoor Object Navigation for the Visually Impaired via YOLOv10 with Neighbor Coordinates and C2FCIB Attention Mechanism. *IEEE Access*. DOI: 10.1109/ACCESS.2025.3612482
[18] Gupta, M. A. S., Raj, M., Kumar, A., & Tripathi, D. (2026). AI-Powered Object Detection and Feedback System for the Visually Impaired. *Data Science and Applications: Proceedings of ICDSA 2025, Volume 2*, *2*, 422.
[19] Kini, K. K., Sreesha, K. R., Barath, M., Poojary, V., & Mishra, S. P. (2025, November). Design and Implementation of AI-Enhanced Smart Device for Assisting the Visually Challenged. In *2025 9th International Conference on Computational System and Information Technology for Sustainable Solutions (CSITSS)* (pp. 1-6). IEEE. DOI: 10.1109/CSITSS67709.2025.11295184
[20] Alashjaee, A. M., AlEisa, H. N., Darem, A. A., & Marzouk, R. (2025). A hybrid object detection approach for visually impaired persons using pigeon-inspired optimization and deep learning models. *Scientific Reports*, *15*(1), 9688.
[21] GILLANI, S. M. Z. R., BANGASH, J. I., & ATTA, M. N. (2025). Enhancing Smart Outdoor Object Navigation for the Visually Impaired via YOLOv10 With Neighbor Coordinates and C2FCIB Attention Mechanism.
[22] Abadi, R. F., Pratama, T. Y., Asmiati, N., Devi, A. A. K., Yuwono, J., Permana, D. S., & Bahrudin, F. A. (2025). "Demata 2.0": An On-Device AI Assistive Technology for the Visually Impaired Integrating YOLOv10 and OCR. *Advance Sustainable Science Engineering and Technology*, *7*(4), 02504026-02504026.
[23] Hui, S. S., Chong, L. B., & Xuen, D. C. Y. (2025, September). Modular Real-Time Mobile Assistive Applications for the Visually Impaired: Enhancing Currency and Document Recognition. In *2025 6th International Conference on Artificial Intelligence and Data Sciences (AiDAS)* (pp. 240-245). IEEE. DOI: 10.1109/AiDAS67696.2025.11213562
[24] Poovannapoikayil, A. V. (2025). *Indian Sign Language Detection and Translation using Deep Learning and Text-to-Speech* (Doctoral dissertation, Dublin, National College of Ireland).
[25] Dang, Z. (2025). YOLO-Improved Lightweight Hybrid Models for Real-Time Blind Path Detection.
[26] Ji, H., Mendonça, I., & Aritsugi, M. (2025). Multi-Scene Dataset and Object Detector for Outside Blind Individual Identification. *IEEE Access*, *14*, 1423-1438.
[27] Sachin, A., Penukonda, A., Naveen, M., Chitrapur, P. G., Kulkarni, P., & BM, C. (2025, June). NAVISIGHT: A Deep Learning and Voice-Assisted System for Intelligent Indoor Navigation of the Visually Impaired. In *2025 3rd International Conference on Inventive Computing and Informatics (ICICI)* (pp. 848-854). IEEE. DOI: 10.1109/ICICI65870.2025.11069837
[28] Li, K., Liang, J., Yin, Y., Fan, X., Ma, C., & Xie, Y. (2025, May). Design of a Raspberry Pi 4B-Based Guide Robot with YOLO-v10 Integration. In *2025 IEEE 7th International Conference on Communications, Information System and Computer Engineering (CISCE)* (pp. 676-680). IEEE. DOI: 10.1109/CISCE65916.2025.11065537
[29] Rastogi, U., Mahapatra, R. P., & Kumar, S. (2025). Advanced gesture recognition in Indian sign language using a synergistic combination of YOLOv10 with Swin Transformer model. *Scientific Reports*, *15*(1), 32894.
[30] Kalaiarasi, P., Naveen, K., Kumar, M. V. A., Adhitya, V., & Lavanya, K. (2025, September). A Real-Time Vision-based Fall Detection System using YOLOv10 and Deep Learning for Ensuring Elderly and Patient Safety. In *2025 6th International Conference on Electronics and Sustainable Communication Systems (ICESC)* (pp. 897-901). IEEE. DOI: 10.1109/ICESC65114.2025.11212604

[31] Bonnin, R., Delrieux, C., & Piccoli, M. F. (2025). Deep Learning-Based Visual Aid for Low Vision. *IEEE Embedded Systems Letters*. DOI: 10.1109/LES.2025.3548959

[32] Patel, D., Singh, D. P., Kour, P., Chhabra, P., Swain, D., Mahajan, S., ... & Vasa, N. (2025, June). An Integrated Framework for Real-Time Object Detection and Speech-Driven Interaction: Advancing Multimodal Human-Like Intelligence. In *International Conference on Data Analytics & Management* (pp. 329-340). Cham: Springer Nature Switzerland.

[33] Saha, K., Ejaz, M. S., & Debnath, S. (2024, December). A Smart Shopping Assistant at Super Shop for Visually Challenged Persons Utilizing Machine Learning Approach. In *2024 International Conference on Recent Progresses in Science, Engineering and Technology (ICRPSET)* (pp. 1-4). IEEE. DOI: 10.1109/ICRPSET64863.2024.10955876

[34] Shafique, S., Bailo, G. L., Gori, M., Sciortino, G., & Del Bue, A. (2025). Enhancing Mobility for the Blind: An AI-Powered Bus Route Recognition System. *Algorithms*, *18*(10), 616.

[35] Serdechnyi, V., Barkovska, O., Kovalenko, A., Havrashenko, A., & Martovytskyi, V. (2025). Research on machine learning methods for detecting objects in difficult shooting conditions. *Radioelectronic and Computer Systems*, *2025*(2), 64-77.

[36] Labhaniya, S., Parmar, J., Pandey, S., Katkade, D., & Mathur, D. (2025, November). A Survey of Machine Learning Techniques for Enhancing Scene Description in Assistive Systems. In *2025 Eighth International Conference on Image Information Processing (ICIIP)* (pp. 407-414). IEEE. DOI: 10.1109/ICIIP68302.2025.11346259

[37] Sriram, S., Aburvan, P., TP, A. K., Vijayaraj, N., & Murugan, T. (2025). Enhanced yolov10 framework featuring dpam and dalsm for real-time underwater object detection. *IEEE Access*, *13*, 8691-8708. DOI: 10.1109/ACCESS.2025.3527315

[38] An, H., Park, W., Liu, P., & Park, S. (2025). Mobile-AI-based docent system: Navigation and localization for visually impaired gallery visitors. *Applied Sciences*, *15*(9), 5161.

[39] Ayush, K., Choubey, A. P., Anand, M. S., Bhar, A., Sudheendran, S., & Dawar, A. (2025, December). SafeRoute: A Survey on GPS-Guided Assistive Sticks for Enhanced Mobility and Safety. In *2025 4th International Conference on Automation, Computing and Renewable Systems (ICACRS)* (pp. 592-599). IEEE. DOI: 10.1109/ICACRS67045.2025.11324314

[40] Moosmann, J., Bonazzi, P., Li, Y., Bian, S., Mayer, P., Benini, L., & Magno, M. (2024, September). Ultra-efficient on-device object detection on ai-integrated smart glasses with tinyissimoyolo. In *European Conference on Computer Vision* (pp. 262-280). Cham: Springer Nature Switzerland.

[41] Li, B., Zhang, R., & Xu, X. (2025). SMC-YOLO: efficient object detector for underwater small sonar target. *Journal of Real-Time Image Processing*, *22*(4), 145.

[42] Gandham, R. (2025). *SMARTGUIDE: Revolutionizing the depth and dependability of Vision-Impaired Navigation* (Doctoral dissertation, Virginia Tech).

[43] Al Duhayyim, M. (2025). Internet of things enabled indoor activity monitoring for visually impaired people with hybrid deep learning and optimized algorithms for enhanced safety. *Scientific Reports*, *15*(1), 35438.

[44] Song, Y., Li, Z., Kang, Y., Li, G., & Xia, H. (2025). An End-to-End Large Model Framework of Wearable Augmented Vision Device for the Visually Impaired. *IEEE Transactions on Automation Science and Engineering*. DOI: 10.1109/TASE.2025.3602402

[45]M Kaliappan, E Mariappan, MV Prakash, B Paramasivan, Load Balanced Clustering Technique in MANET using Genetic Algorithms.. Defence Science Journal 66 (3), 251-258.

[46]M Sivaram, M Kaliappan, S J Shobana, Prakash, V Porkodi Secure storage allocation scheme using fuzzy based heuristic algorithm for cloud, Journal of Ambient Intelligence and Humanized Computing, pp.1-9

[47]Vimal, S., Robinson, Y. H., Kaliappan, M., Vijayalakshmi, K., &Seo, S. (2021). A method of progression detection for glaucoma using K-means and the GLCM algorithm toward smart medical prediction. The Journal of Supercomputing, 77(1), 1–17. https://doi.org/10.1007/s11227-020-03268-0

[48]Kaliappan M, Guruprakash B, Rajalakshmi, J. Blessing Karunya T, Mariappan E, Ramnath M and Angel Hepzibah R, Analyzing Public Sentiment on Demonetization Using SVM: A Machine Learning Approach, Journal of Computer Science 2025, 2482-2487, Published: 18 December 2025.