International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 01 | Jan - 2026 SJIF Rating: 8.586 ISSN: 2582-3930

AI-Driven Object and Scene Recognition System for Blind Assistance

ABHILASH T S, CHIDANANDAYYA, PAVITRA S, VISHAL SHIVASHARAN BILUR,

Prof. ANIL KUMAR

COMPUTER SCIENCE & ENGINEERING
AMRUTA INSTITUTE OF ENGINEERING & MANAGEMENT SCIENCES

Abstract:

The rapid advancement of Internet of Things (IoT) and Artificial Intelligence (AI) technologies has enabled the development of intelligent systems capable of perceiving and interpreting real-world environments. This project presents an Intelligent Visual Assistant that integrates an ESP32-CAM module with a server based AI object detection framework to provide real-time visual understanding and voice-based feedback. The ESP32-CAM captures images and transmits them wirelessly to a Flask-based server, where the images are processed using the YOLOv8 object detection algorithm to identify objects present in the scene. The detected information is converted into audible speech using a Text-to-Speech (TTS) engine and delivered to the user through a speaker or Bluetooth-enabled device.

The proposed system adopts a hybrid edge–server architecture that balances low-cost hardware with high-performance AI processing. Experimental results demonstrate reliable object detection accuracy with low response latency under local network conditions, making the system suitable for real-time applications. The solution is particularly beneficial for assistive technologies for visually impaired individuals, smart surveillance etc.

Key Words: ESP32-CAM, Object Detection, YOLOv8, Internet of Things (IoT), Text-to-Speech (TTS), Real-Time Image Processing, Assistive Technology, Embedded Systems, Wireless Communication.

1. INTRODUCTION

This project presents an intelligent AI-powered visual assistant built using the ESP32-CAM, Flask server, YOLOv8 object detection model, and Text-to-Speech (TTS) technology. The system captures real-time images through the ESP32-CAM and sends them to a local server over Wi-Fi, where advanced machine learning algorithms analyse the image and identify objects present in the scene. The detection results are then converted into voice feedback, which is played through a connected speaker, enabling hands-free interaction. Designed to be low-cost, portable, and efficient, this solution aims to support visually impaired users while also offering applications in security, monitoring, and automation. By integrating IoT with AI, the project demonstrates how edge devices and server-based AI models can work together to deliver fast, accurate, and user-friendly real-time object detection.

The rapid advancement of the Internet of Things (IoT) and Artificial Intelligence (AI) has transformed how intelligent systems are built and deployed across various industries. Modern embedded devices are no longer limited to simple sensing and control; instead, they are increasingly capable of communicating, collaborating, and performing complex tasks with the support of cloud or local computing infrastructures. Within this technological revolution, the ability for machines to perceive and understand their surroundings visually has become one of the most impactful developments.

2. Body of the Paper

The body of the paper presents the technical details, methodology, system design, implementation, and results of the proposed intelligent visual assistance system. This section is divided into well-structured subsections to clearly explain each stage of the system. References to sections are made using section numbers for clarity and consistency.

2.1 System Overview

The proposed system is an IoT-based intelligent visual assistant designed to perform real-time object detection and provide voice feedback to users. The system integrates an ESP32-CAM module, a Flask-based local server, the YOLOv8 (You Only Look Once version 8) object detection algorithm, and a Text-to-Speech (TTS) engine. The ESP32-CAM captures images and transmits them over a local Wi-Fi network to the server, where image processing and object detection are performed. Detected objects are converted into speech and played through a speaker, enabling auditory feedback.

2.2 Hardware Components

The primary hardware component used in this project is the ESP32-CAM (Espressif Systems) module, which combines a microcontroller with an integrated camera. It supports Wi-Fi communication and is suitable for low-cost embedded vision applications. An external speaker is connected via an I²S (Inter-IC Sound) Digital-to-Analog Converter (DAC) to deliver audio feedback. A stable power supply ensures uninterrupted operation of the system.

2.3 Software Architecture

The software architecture consists of three major layers: data acquisition, processing, and output generation. The ESP32-CAM runs embedded firmware to capture images and send them using HTTP (Hypertext Transfer Protocol) POST requests. The server side is implemented using Flask (Python web framework), which receives the image, decodes it using OpenCV (Open Source Computer Vision Library), and processes it using YOLOv8. The detected object labels are converted into speech using a TTS engine and played back to the user.

2.4 Object Detection Algorithm

Object detection is performed using YOLOv8, a state-of-the-art deep learning model for real-time object detection. YOLOv8 divides the input image into grids and predicts bounding boxes and class probabilities in a single forward pass, enabling fast inference. In this project, the lightweight YOLOv8n (nano) model is used to balance detection accuracy and processing speed. Although larger models such as YOLOv8s offer higher accuracy, YOLOv8n is more suitable for real-time and low-resource environments.

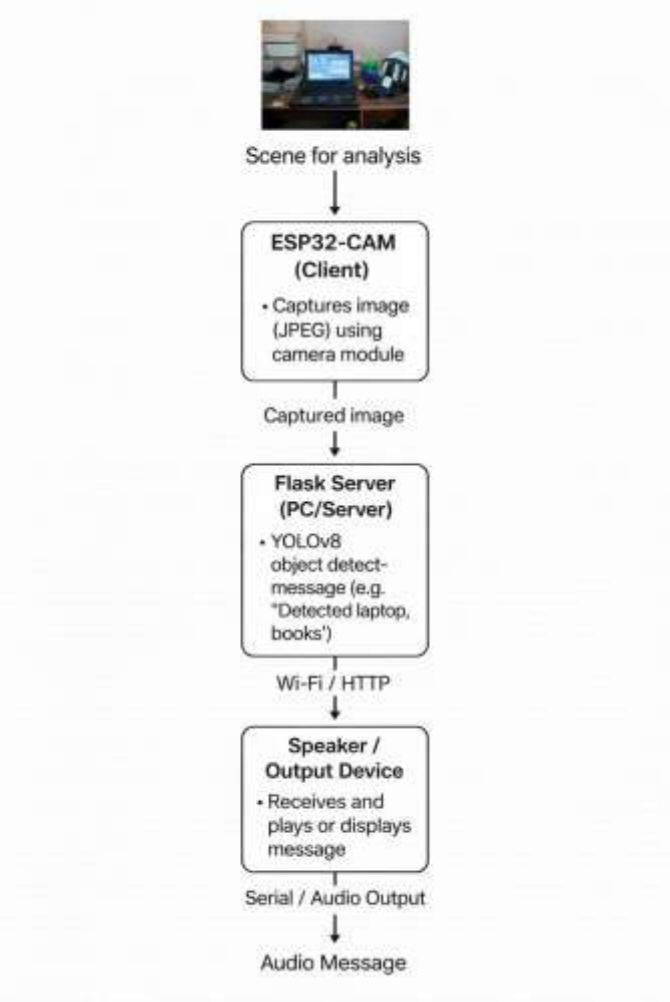
2.5 Data Flow and Communication

The data flow begins with image capture at the ESP32-CAM, followed by wireless transmission to the server via Wi-Fi. As described in Section 2.4, the server processes the image and performs object detection. The resulting object names are sent to the TTS module, which generates an audio output. The feedback is transmitted to the speaker wirelessly or through a wired connection, completing the end-to-end workflow.

2.6 Implementation Details

The firmware for the ESP32-CAM is developed using the Arduino Integrated Development Environment (IDE). The server-side application is written in Python, leveraging libraries such as Flask, OpenCV, and Ultralytics YOLO. The TTS functionality is implemented using a local speech synthesis library to ensure offline operation. The system is tested under various lighting and object conditions to evaluate performance and reliability.

2.7 Block Diagram



The block diagram provides a high-level representation of the Intelligent Visual Assistance System, illustrating the sequence of operations and the interaction between its core components. The process begins with the ESP32-CAM module, which captures real-time images from the environment and transmits them to the server over a Wi-Fi connection.

These images are then received by the Flask Server, where the integrated YOLOv8 object detection model processes the input and identifies objects present in the captured frame. The detected results are forwarded to the Text-to-Speech (TTS) module, which converts the textual labels into audio output. This generated audio feedback is then transmitted via Bluetooth to a speaker or user device, enabling real-time auditory guidance. The diagram clearly outlines this end-to-end flow—from image capture to

intelligent analysis and final voice-based output—highlighting the modular design and seamless coordination between IoT hardware, AI processing, and communication units. This structured block representation demonstrates the efficiency, clarity, and integration of the entire system.

2.8 Output and Result

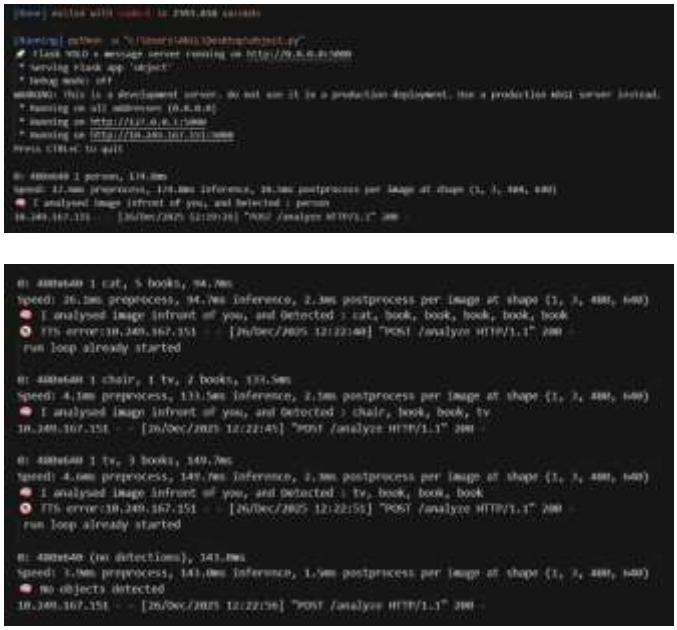


Fig 2.8a:- Real-Time Detection Results Generated by ESP32-CAM and YOLO Model

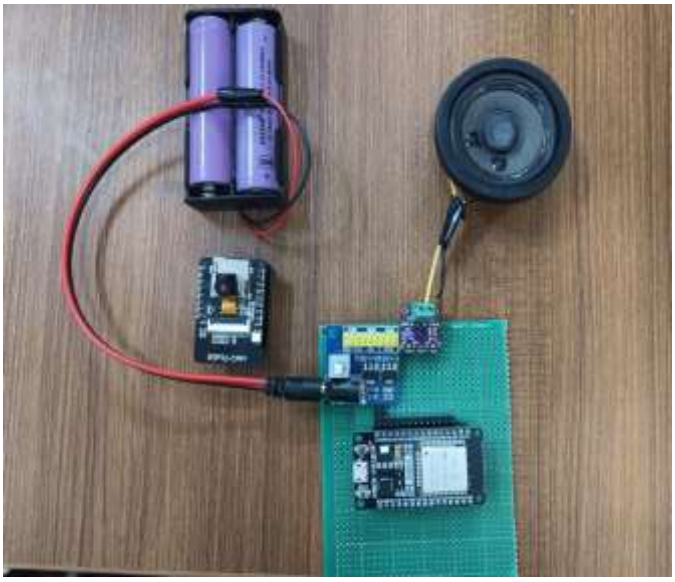
The above screenshots show the runtime output of the proposed system, which integrates an ESP32-CAM module, a Flask-based backend server, YOLO object detection, and Text-to-Speech (TTS) for audio feedback. The outputs confirm successful end-to-end operation of the system. The snapshot shows the real-time execution and output of the proposed ESP32-CAM–based object detection system integrated with a Flask server, YOLO model, and text-to-speech module. The Flask backend is successfully initialized and runs on port 5000, receiving images from the ESP32-CAM through HTTP POST requests. For each incoming image, the YOLO model processes a 480×640 resolution frame and detects objects such as person, bench, cat, laptop, book, TV, chair, mouse, and suitcase, along with the corresponding inference time, which typically ranges from around 90 ms to 220 ms, indicating near real-time performance.

Fig 2.8b:- Serial Monitor and Server Response Output of ESP32-CAM Object Detection System

The snapshot illustrates the real-time operation and output of the proposed ESP32-CAM–based object detection system integrated with a Flask server. After power-on, the ESP32-CAM successfully initializes the camera module, connects to the Wi-Fi network, and obtains a valid IP address, confirming proper hardware and network configuration. The captured images are transmitted to the server via HTTP requests, and the server consistently returns an HTTP response code 200, indicating successful communication and processing.

The server analyzes each received image using the implemented object detection algorithm and identifies multiple objects such as person, cat, laptop, book, mouse, keyboard, chair, TV, and airplane. In some frames, repeated detections of the same object (e.g., multiple books) demonstrate the system's ability to detect multiple instances within a single image.

2.9 Working Model



The implemented Intelligent Visual Assistant system successfully achieved its intended objectives by providing real-time object detection and voice-based feedback using a hybrid IoT and AI architecture. The ESP32-CAM effectively captured images and transmitted them to the Flask server over a local Wi-Fi network.

The server processed the received images using the YOLOv8 algorithm, accurately identifying objects present in the scene. The detected object labels were then converted into audible speech using the Text-to-Speech (TTS) module and delivered to the user through a connected speaker or Bluetooth device. The system demonstrated smooth end-to-end functionality from image capture to audio output, validating the feasibility of the proposed design.

3 CONCLUSIONS

The proposed Intelligent Visual Assistant system successfully integrates IoT hardware, wireless communication technologies, and advanced AI-based object detection into a unified, functional prototype. By leveraging the ESP32-CAM for low-cost image acquisition and offloading the computationally intensive YOLOv8 processing to a local Flask server, the project demonstrates a practical hybrid architecture that balances performance, cost efficiency, and responsiveness. The system also incorporates Text-to-Speech (TTS) and Bluetooth communication, enabling real-time audio feedback that significantly enhances accessibility for users, especially those with visual impairments.

Through systematic design, implementation, and testing phases, the project validates that affordable embedded systems can be effectively combined with modern machine learning algorithms to deliver real-time intelligent assistance. The achieved results—fast inference, accurate object detection, and seamless multimodal output—highlight the system's usability in diverse domains such as security, automation, and industrial monitoring. Overall, the project establishes a strong foundation for future enhancements, including mobile app integration, improved AI models, and broader real-world deployment, emphasizing its potential societal and technological impact.

ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to the Department of Computer Science and Engineering AMRUTA INSTITUTE OF ENGINEERING & MANAGEMENT SCIENCES, BIDADI , BENGALURU -562109 for providing the laboratory facilities and continuous support throughout the project.

REFERENCE

[1] T. Nguyen et al., "Secure IoT Data Transmission," IEEE Transactions on IoT, 2023.
Covers best practices for sending sensor data securely. Guided design considerations for HTTP communication.

[2] G. Jocher et al., "YOLOv8: Ultralytics Official Documentation," 2023.
Provides architecture, training methods, and deployment workflows. Used to implement the object detection pipeline on the Flask server.

[3] A. Rosebrock, "PyImageSearch — Object Detection Tutorials," PyImageSearch, 2022.
Explains practical image preprocessing and computer vision techniques. Helped improve server-side image decoding and handling.

[4] K. Patel et al., "Lightweight Deep Learning Models for IoT," IEEE IoT Magazine, 2022.
Highlights the advantages of compact models for edge devices. Justifies using YOLOv8n for faster inference.

[5] A. Paul and S. Sharma, "AI-Based Surveillance using Low-Cost Hardware," IEEE Conference on Smart Systems, 2022.
Discusses combining microcontrollers with server-based AI. Inspired the hybrid architecture of the project.

[6] S. Gupta, "Real-Time Multimedia Processing in IoT Devices," IEEE Multimedia, 2021.
Explores limitations of camera modules like ESP32. Reinforced the design choice to offload computational tasks.

[7] M. Srivastava, "Voice Assistive Systems Using TTS," IEEE Transactions on Consumer Electronics, 2021.
Reviews various text-to-speech solutions. Guided the implementation of pyttsx3 for local speech synthesis.

[8] S. Shaikh and R. Pawar, "Camera-Based Assistive Systems for the Visually Impaired," IEEE Sensors Journal, 2021.
Presents techniques for voice-based guidance using computer vision. Influenced the accessibility-focused design of the system.

[9] M. Z. Hasan et al., "IoT-Based Smart Monitoring Systems," IEEE IoT Journal, 2021.
Discusses IoT architectures for real-time data acquisition and control. Helps align the project with industry-standard IoT frameworks.

[10] Espressif Systems, "ESP32-CAM AI Thinker Module Datasheet," 2021.
Describes camera specifications, pinout, and hardware limitations critical for integration. Forms the hardware foundation of the project.