I.J. Information Technology and Computer Science, **2026, 4, 50-67**
Published Online on August 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2026.04.04

Edge AI-Based Object Detection via Voice Recognition with an LLM-Based Emotional Assistant for Elderly Care Robots

Sarra Ben Halima*
Automation Research Laboratory (LARA), LR11ES18, National Engineering School of Tunis, University of Tunis El Manar, 1002 Tunis, Tunisia
E-mail: sarah.benhlima@enit.utm.tn
ORCID iD: https://orcid.org/0009-0002-2195-8800
*Corresponding author

Faten Ben Abdallah
Automation Research Laboratory (LARA), LR11ES18, National Engineering School of Tunis, University of Tunis El Manar, 1002 Tunis, Tunisia
E-mail: faten.benabdallah@enit.utm.tn
ORCID iD: https://orcid.org/0000-0002-2449-2199

Joseph Haggege
Automation Research Laboratory (LARA), LR11ES18, National Engineering School of Tunis, University of Tunis El Manar, 1002 Tunis, Tunisia
E-mail: joseph.haggege@enit.utm.tn
ORCID iD: https://orcid.org/0000-0002-6068-0568

Received: February 16, 2026; Revised: April 29, 2026; Accepted: May 27, 2026; Published: August 8, 2026

Abstract: This paper presents a fully integrated, real-time assistive system that combines voice-based object recognition with a generative conversational interface, specifically designed to enhance elderly care via edge AI deployment. The proposed framework enables intuitive human–robot interaction in domestic environments by fusing natural language understanding, optimized visual detection, and on-device generative AI. Voice commands are processed via a speech-to-text pipeline using the Google Web Speech API. Keyword extraction triggers object detection using a quantized YOLOv8n model, accelerated by TensorRT with FP16 inference on an NVIDIA Jetson Nano. In parallel, a locally deployed generative AI assistant, executed entirely on-device, provides empathetic dialogue to support social engagement and emotional well-being. The proposed system adopts a hybrid edge architecture in which object detection, robot control, and LLM-based dialogue generation are executed on-device, while speech-to-text transcription relies on a cloud-based service. This generative interface is implemented as an LLM-based Emotional Assistant. The system achieves 13 FPS with an inference latency of 70 ms for object detection, 94.3% speech recognition accuracy, and an F1-score of 0.69 at a 0.5 confidence threshold. All AI components are executed on-board, preserving privacy for on-device processing while maintaining real-time responsiveness. Experimental validation confirms the effectiveness of deploying multimodal AI, including generative models, on resource-constrained hardware. This work lays the foundation for autonomous, voice-guided care robots that not only assist in locating objects but also engage users socially, promoting greater autonomy and quality of life for older adults.

Index Terms: Elderly Care, Edge AI, NVIDIA Jetson Nano, Speech Recognition, Generative AI, Object Detection, Embedded Systems, Assistive Robotics, LLM-based Emotional Assistant.

1. Introduction

The integration of Artificial Intelligence (AI) into embedded systems has revolutionized assistive robotics, enabling real-time decision-making and multimodal interaction in low-power environments. This technological shift is particularly impactful for elderly care, where autonomy, reliability, and emotionally intelligent interfaces are essential. In this context,

we present the design and implementation of a novel embedded robotic system for elderly assistance, combining voice-controlled object recognition with a locally executed generative conversational agent. The entire platform is deployed on an NVIDIA Jetson Nano, enabling real-time interaction through speech and vision without dependence on cloud services.

With the global population aging rapidly, elderly individuals increasingly face challenges related to mobility limitations, cognitive decline, and sensory impairments. These factors can hinder daily tasks and contribute to feelings of isolation and decreased quality of life. According to the World Health Organization (WHO), over 20% of people aged 60 and above suffer from mental health issues linked to loneliness and reduced self-efficacy [1]. The National Academies of Sciences further highlight the need for scalable technological interventions to mitigate social isolation and enhance emotional well-being [2].

To address this pressing societal need, our system offers a dual-function solution that provides both practical assistance and empathetic interaction. First, a natural language speech interface allows users to request object recognition using intuitive voice commands (e.g., "Where is my bottle?"). These commands are processed through a speech-to-text pipeline and matched against class labels of the YOLOv8n object detection model, enabling real-time visual localization of the requested object. Second, we integrate an emotionally supportive generative AI assistant, *CompanionCare*, based on LLaMA 3 and deployed locally via Ollama. This chatbot interprets user intent, engages in personalized dialogue, and offers affective support through voice, promoting companionship and reducing feelings of loneliness.

To meet the requirements of embedded AI deployment, we utilize the NVIDIA Jetson Nano—a compact, energy-efficient computing platform equipped with a 128-core CUDA-enabled GPU. This hardware offers a compelling balance between computational power and energy consumption, making it particularly suitable for real-time AI inference in resource-constrained environments.

Our key contributions are summarized as follows:

- We design a speech-driven perception pipeline that uses keyword extraction to trigger object recognition via YOLOv8n, enabling intuitive voice-controlled interaction.
- We introduce *CompanionCare*, a generative AI-based chatbot assistant capable of engaging in emotionally adaptive conversation, synthesized through real-time Text-to-Speech (TTS).
- We implement a fully embedded multimodal perception pipeline on the NVIDIA Jetson Nano, combining real-time speech recognition and object detection. To ensure suitability for edge deployment, we apply post-training FP16 quantization and TensorRT-based acceleration, optimizing both latency and memory usage for resource-constrained environments.
- We conduct a thorough evaluation of recognition accuracy, response latency, and conversational coherence to validate the feasibility of our approach in real-world assistive scenarios.

This work distinguishes itself from existing socially assistive robotic (SAR) platforms by offering a fully embedded, privacy-preserving solution optimized for low-resource environments. While prior systems such as RAMCIP and Care-O-bot [3, 4] have demonstrated functional assistance, they often rely on cloud infrastructure, scripted interactions, or lack emotional responsiveness. In contrast, our system delivers autonomous operation and multimodal engagement, supporting natural interactions even in offline contexts.

In summary, this paper introduces an edge-deployable robotic platform that integrates real-time voice and vision processing with emotionally intelligent dialogue generation, targeting the dual challenges of physical assistance and emotional support for elderly users. Its modular and resource-efficient architecture lays the foundation for future extensions including gesture-based control, navigation, or robotic manipulation in home-care environments.

The remainder of this paper is structured as follows: Section 2 reviews related work in assistive robotics and embedded AI. Section 3 presents the system architecture and components. Section 4 describes the simulated testing and interaction logic. Section 5 details the hardware deployment and optimization strategies. Section 6 analyzes the experimental performance. Finally, Section 7 concludes with a discussion of the findings and future perspectives.

2. State of the Art

Recent advances in embedded AI have driven the emergence of intelligent assistive systems tailored for elderly and visually impaired populations [5]. These systems typically combine computer vision, voice interaction, and deep learning to enable context-aware and responsive robotic behavior.

To enhance human-robot communication, EchoVision [6] introduced a real-time assistive platform using YOLOv8 for object detection, speech-to-text conversion, and multilingual voice feedback. Though originally developed for visually impaired users, the system's interaction pipeline is relevant to broader assistive robotics applications.

Other systems such as ENRICHME [7] and Pepper [8] demonstrate how interactive capabilities can be integrated into home robots to support user engagement. However, studies on acceptance and adoption such as Fracasso et al. [9] and Cavallo et al. [10] show that long-term integration depends on personalization, usability, and reliability—dimensions our work directly addresses.

A large body of work has also focused on visual assistance systems using deep learning and voice interfaces. For example, Visio-Voice [11] and Envision [12] employ YOLO-based object detection coupled with auditory feedback for

navigation and object localization. The Trinetra app [13], V-Eye system [14], and Sight-to-Sound interface [15] further leverage embedded vision and sound to enhance spatial awareness. These solutions are typically low-cost and optimized for edge devices, making them suitable for real-time use in home environments.

Several studies have explored speech-enabled object detection and AIoT frameworks. Works like AI-SenseVision [16], Smart Stick systems [17], and AI-based navigation assistants [18, 19] integrate deep learning with wearable sensors and IoT components for robust performance in unstructured environments. Meanwhile, haptic feedback approaches [20] and virtual assistant integration [21] offer alternative interfaces for interaction.

In this line of research, Djinko et al. [22] proposed a video-based system that combines YOLOv7 and Google Speech API to locate previously recorded objects through voice queries. While efficient in retrieving visual evidence, their approach lacks real-time responsiveness and robotic interaction. Gygli et al. [23] explored fast object annotation via speech for large-scale datasets like COCO and ILSVRC, achieving substantial reductions in annotation time. Their speech-click interface, though not designed for robots, underlines the efficiency of voice as a modality for semantic interaction.

From a practical standpoint, Ramesh et al. [24] presented a voice assistant for visually impaired users using ESP32 and ChatGPT integration, focusing on low-cost natural language processing. Their system emphasizes speech interaction and IoT control but does not include object detection or robotic mobility. In contrast, Miyata et al. [25] developed an edge-AI-based mobile robot on Jetson Nano that combines voice and object recognition to locate specified targets. Their system leverages Robot Operating System (ROS) and YOLO but remains limited to predefined commands and lacks semantic understanding.

In parallel, Gordeev et al. [26] proposed an autonomous mobile robot equipped with an AI-based perception system built around the Jetson Nano platform. Their robot integrates dual LiDAR sensors fused into a single point cloud, a deep convolutional neural network (DCNN) for object detection, and a Dlib tracker for object tracking. In addition to low-light visual capabilities using an IR-assisted IMX219 camera, the robot includes fuzzy logic-based motor control and employs Simultaneous Localization and Mapping (SLAM) via Google Cartographer. Notably, they combine A* path planning and ROS-based modular architecture to enable autonomous navigation and real-time scene understanding. While their system addresses search and mapping tasks in complex environments, it does not focus on speech-based interaction or elderly care scenarios, which our work specifically targets through voice-triggered object recognition.

Despite these advances, most existing solutions focus on either visual assistance or smart home integration. Few systems aim to combine robust object recognition, voice-controlled interaction, and real-time operation in a compact edge-based architecture. Our work addresses this gap by introducing a unified platform for elderly users that integrates voice- controlled object detection with onboard AI processing on Jetson Nano, offering a responsive and efficient assistive solution.

To better position our contribution within the existing literature, Table 1 provides a comparative summary of recent embedded assistive systems. It highlights the key features of each solution, including voice recognition method, object detection model, hardware platform, real-time capability, and targeted context. As shown, very few systems simultaneously offer voice-controlled object detection, real-time edge inference, and full autonomy on Jetson Nano—precisely the core focus of our proposed framework.

Table 1. Comparison of recent embedded AI systems for voice-based object recognition

System	Voice Recognition	Object Detection	Hardware form Plat-	Context/Target Users
EchoVision [6]	Google API + multilingual TTS	YOLOv8	Laptop (non-embedded)	Visually impaired users
ENRICHME [7]	Contextual dialogue	Non-CNN context detec- tion	TIAGo robot	Elderly users
Visio-Voice [11]	Simple voice commands	YOLOv4 + audio feedback	Raspberry Pi (limited)	Visual navigation
Djinko et al. [22]	Google API (voice queries)	YOLOv7 (retrieval only)	PC with webcam	Voice-based video search
Miyata et al. [25]	Static voice commands	YOLO + ROS integration	Jetson Nano	Home assistance
Gordeev et al. [26]	Not specified	DCNN + Dlib tracker	Jetson Nano	Autonomous search robot
Our System	Google Web Speech API + TTS + LLM-Based Emotional Assistant	YOLOv8n + TensorRT (FP16)	Jetson Nano	Voice-guided object recognition and emo- tional support for elderly care

3. Methodology

Fig. 1 illustrates the architecture of the proposed assistive system, designed to support elderly users through voice-commanded object retrieval and interactive dialogue. The system is built around two main components: the user interface, composed of a microphone and a camera, and the robotic platform, which performs perception, decision-making, and actuation tasks. The interaction begins with the user issuing a voice command via the microphone. This input is processed by a Voice Recognition Module, which is trained using the Google Speech Commands Dataset. The recognized command, typically specifying an object of interest, is passed to the Object Detection Module, which processes live video input from

the camera. This module uses a YOLOv8-based neural network, pre-trained on the COCO dataset, to identify and localize the requested object in the scene.

Once the object is detected and localized, its position is sent to the Robot Control System, which plans and executes the necessary actions to navigate toward the object and pick it up. The object is then delivered to the user, completing the retrieval loop.

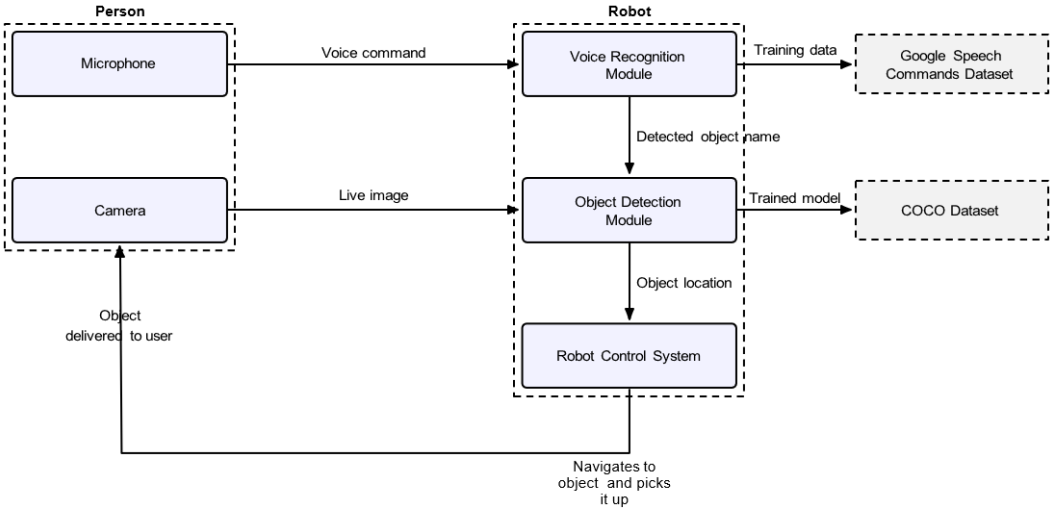

Fig.1. System architecture for voice-commanded object retrieval using audio-visual perception and robot control

In addition to its perception and actuation capabilities, the system is augmented with a generative conversational assistant, powered by the LLaMA3 model and running locally via Ollama. This component enables natural, context-aware dialogue, enhancing user experience through both practical guidance and emotional support.

By tightly integrating voice recognition, computer vision, and robotic control on an embedded platform such as the NVIDIA Jetson Nano, the proposed system demonstrates a compact, affordable, and user-friendly solution for assistive robotics in home environments.

The voice recognition module is implemented using the SpeechRecognition Python library [27], combined with the recognize_google() API that interfaces with Google's Speech-to-Text service [28]. The process begins with capturing the user's spoken request via a connected microphone using the PyAudio library [29]. The recorded audio is then converted into textual form and analyzed to extract object-related keywords (e.g., "bottle", "medicine"). These extracted labels are matched against a predefined vocabulary of detectable objects. The accuracy of recognition depends on ambient noise and articulation clarity, which are critical considerations in elderly use cases.

Once the keyword is extracted, the object detection process begins. A camera module continuously captures live frames, which are processed using a pretrained YOLOv8 model deployed on the Jetson Nano. Inference is accelerated via TensorRT with FP16 quantization, enabling the system to achieve real-time processing between 7 and 10 FPS. The detected objects are compared against the extracted keyword from the user's speech. When a match is found, the system visually highlights the corresponding object with bounding boxes and confidence labels, completing the voice-to-vision interaction cycle. The output is rendered on-screen for visual feedback to the user.

The implementation followed a two-phase methodology. In the first phase, a software-based approach was adopted in which all components (including voice recognition, object detection, and conversational generation) were independently developed and tested within a controlled software environment. Integration was performed using Python-based tools under Windows, ensuring end-to-end validation of the software stack. In the second phase, an edge deployment strategy was applied. The complete pipeline was migrated to the Jetson Nano platform, with particular attention given to resource efficiency and inference latency. Peripheral devices such as a USB microphone and camera were connected and configured to support real-time audio-visual data acquisition.

The Jetson Nano offers an excellent balance between compute performance (472 GFLOPS) and power consumption (< 10 W), making it well-suited for embedded AI applications. It provides native support for AI deployment frameworks such as PyTorch, OpenCV, and ONNX, and it is compatible with NVIDIA JetPack, which includes CUDA and cuDNN for GPU acceleration. The YOLOv8 model was exported to ONNX format and optimized with TensorRT, significantly reducing inference time. JupyterLab was used to manage development and prototyping. Deployment took place under Ubuntu Linux, ensuring compatibility with embedded toolchains and stable execution. The system's modularity also allows for future improvements and additional sensors to be integrated.

In summary, the methodology demonstrates a scalable and energy-efficient approach to embedded AI system development. By coupling speech and visual recognition, and optimizing for low-latency deployment on edge hardware, this system provides a functional and accessible solution for real-time assistance in elderly care contexts.

4. Software Deployment

The Software phase constituted the initial validation step of our speech-driven object detection system. It was conducted entirely on a host machine running Windows, which facilitated flexible debugging and rapid prototyping. This phase involved the separate development and validation of two core modules: speech recognition and object detection.

4.1. Speech Recognition

The speech recognition process begins with the initialization of a recognition module capable of capturing and interpreting voice commands from a microphone. To ensure robustness in real-world environments, the system first performs an ambient noise calibration. This step allows the recognition engine to dynamically adjust its sensitivity threshold, reducing the impact of background noise on subsequent voice analysis.

Once the system is calibrated, the user is prompted to speak, and a short audio segment—typically lasting five seconds—is recorded. This segment serves as the voice input to be analyzed and converted into text. The captured audio is then processed by the Voice Recognition Module, which uses the Google Web Speech API to perform transcription into natural language.

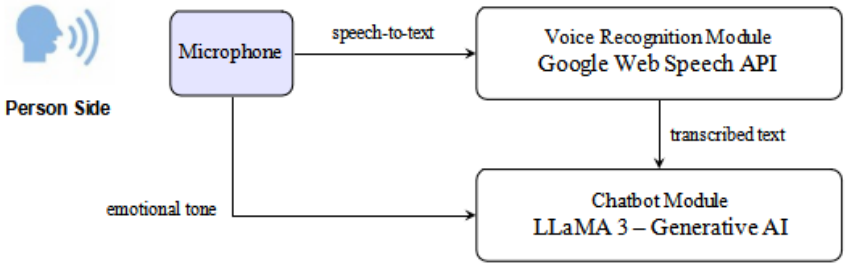

Fig.2. Architecture overview: voice interaction and emotional feedback pipeline using Google Web Speech API and LLaMA 3

Fig. 2 illustrates the full interaction pipeline of the assistive robot system. The process starts with the user speaking a natural language command. This audio input is captured by the microphone and processed by the Voice Recognition Module, which performs transcription using the Google Web Speech API.

The resulting transcribed text is then forwarded to a Generative AI Chatbot Module powered by LLaMA 3 via the Ollama framework. This module interprets the user's intent and generates context-aware responses. Importantly, emotional cues such as tone and pitch are directly extracted from the user's voice input by the chatbot to adapt responses accordingly—an essential feature for elderly care and emotional companionship.

To ensure system reliability, multiple exception-handling mechanisms are implemented:

- *Network fallback:* If the transcription service is unavailable, the system gracefully notifies the user.
- *Voice quality check:* If the input is unintelligible or silent, the system avoids misinterpretation and false trigger.

The proposed voice-driven chatbot system operates through a sequence of interconnected modules to enable real-time interaction. First, the user provides input by speaking naturally to the robot. This spoken input is captured and transmitted by the local system to the natural language processing (NLP) module. The NLP component then analyzes the input, extracting its semantic meaning and emotional tone to determine an appropriate response. It may consult a local knowledge base or previously stored data to enrich its understanding. Based on this context, the chatbot engine generates a coherent and emotionally sensitive response tailored to the user's needs. Finally, the response is delivered back to the user either as synthesized speech or textual feedback, completing the conversational loop in a natural and empathetic manner.

These mechanisms ensure robustness and stability of the overall system, which is particularly important for assistive applications requiring continuous and reliable operation.

Algorithm 1 describes a robust speech-to-response pipeline that enables natural interaction between a user and an embedded assistive system. The process begins with the initialization of essential Python libraries, including modules for speech recognition, text-to-speech, audio playback, and communication with the LLaMA 3 model via Ollama. After importing the required tools, the system performs ambient noise calibration to adapt to environmental conditions, improving transcription accuracy. The user's voice input is then captured through a microphone and transcribed using the Google Web Speech API. The resulting text is semantically analyzed to extract the user's intent and emotional tone, which guides the next phase—response generation. A contextual and emotion-aware prompting reply is produced locally using the

LLaMA 3 generative model. Emotional awareness in the current system is achieved through prompt-level conditioning of the LLM based on detected keywords and interaction context, without explicit emotion classification or affective model validation. This response is synthesized into speech using gTTS and played back via the system's speaker. The algorithm integrates robust error handling to manage API unavailability, unintelligible speech, or unexpected runtime errors, ensuring reliability and smooth operation in real-time, especially in assistive scenarios.

Algorithm 1: Speech-to-Response Pipeline using Google API and LLaMA 3 via Ollama

Input: User's spoken input
Output: Recognized text or spoken response
Data: Microphone audio stream; Google Web Speech API; LLaMA 3 via Ollama
/* Import and Initialization { Required Libraries */
Import the following Python libraries:
- speechrecognition: voice capture and transcription
- gTTS: speech synthesis from text
- pygame: audio playback
- requests: local communication with LLaMA 3
- time, os, random, io: utility and system functions Initialize

recognizer: recognizer ← sr.Recognizer()
Function recognizespeech():
  /* Step 1: Microphone Input and Noise Calibration */
  Open microphone as input source
  Print "Adjusting for ambient noise..."
  recognizer.adjustforambientnoise()
  Print "Say something!"
  try
    /* Step 2: Capture and Transcribe with Google Web Speech API */
    Record audio for 5 seconds → audio
    Print "Recognizing..." text
    ←recognizer.recognizegoogle(audio) Print recognized text
    /* Step 3: Natural Language Understanding */
    Extract intent and emotional tone from text
    /* Step 4: Query Local Knowledge */
    If applicable, query knowledge base or memory
    /* Step 5: Generate Response using LLaMA 3 */
    response← llama3.generate(text)
    /* Step 6: Output via TTS and Audio Playback */
    Convert responseto audio using gTTS
    Play the response using pygame

  end
  catch
    RequestError
  end
  Print "API unreachable."
  catch
    UnknownValueError
  end
  Print "Speech not understood."
  catch (all)
    Print error message
  end
Call recognizespeech()

The speech recognition module was evaluated using the Google Web Speech API. As shown in Table 2, the system achieved an accuracy of *94.3%* in quiet indoor environments. This high recognition rate ensured reliable keyword extraction from spoken commands, allowing the system to robustly trigger object detection routines in real time. Additionally, the local deployment of the LLaMA 3 model via Ollama was successfully tested, occupying 6.2 GB of memory and utilizing 14% of the CPU and 86% of the GPU. These metrics confirm the feasibility of on-device inference, with minimal latency and efficient resource usage.

The results confirm the suitability of cloud-based APIs for embedded voice interfaces when latency and connectivity conditions are acceptable, while also highlighting the complementary benefit of local LLM deployment for privacy and responsiveness. Despite the simplicity of integration, the overall system performance remains competitive with more complex architectures. Moreover, real-time interaction was preserved without significant delay, demonstrating the practical viability of using voice as a natural interface for visually impaired or elderly assistance systems.

Table 2. Speech and chatbot system performance

Metric	Value
Speech Recognition Accuracy	94.3%
Integration Mode	Real-Time
LLM Deployment	6.2 GB, 14% CPU / 86% GPU

4.2. Object Detection

The object detection pipeline was constructed using the lightweight YOLOv8n model from the Ultralytics framework. The YOLOv8n model used in this work is pretrained on the COCO dataset and is employed directly for inference; no additional training or fine-tuning is performed. Initial testing was performed on static images using OpenCV to verify the correctness of bounding boxes, label assignment, and confidence visualization. After validating the inference process,

the pipeline was extended to real-time video streams using a webcam. Each frame was processed in real-time to detect and annotate objects dynamically.

Algorithm 2: Object Detection in an Image using YOLOv8 and OpenCV

Input: Input image path
Output: Image annotated with detected objects
Data: YOLOv8n model trained on COCO dataset, OpenCV image handling

```
                                                    /* Import libraries and load model */
Import ultralytics.YOLO and cv2
Load YOLOv8n model: model ← YOLO("yolov8n.pt")
                                                    /* Read and process input image */
Read image: image ← cv2.imread(image path)
Run detection: results ← model(image)
                                                    /* Loop through detection results */
foreach result in results do
    foreach box in result.boxes do
        Extract coordinates: x1, y1, x2, y2 ← box.xyxy[0]
        Extract confidence: conf ← box.conf[0]
        Get class index: cls ← int(box.cls[0])
        Compose label: label ← model.names[cls] + confidence
                                                    /* Annotate the image */
        Draw bounding box: cv2.rectangle(...)
        Add label text: cv2.putText(...)
    end
end
                                                    /* Display results */
Show image: cv2.imshow("YOLOv8 Detection", image)
Wait for key press: cv2.waitKey(0)
Close window: cv2.destroyAllWindows()
```

After validating the speech recognition and object detection modules independently, they were integrated into a unified voice-controlled perception system. In this setup, when a user utters a command containing an object keyword (e.g., "bottle", "chair"), the system extracts the keyword via speech recognition and matches it against the class labels returned by the object detection model. If the specified object is present in the camera feed, its bounding box is highlighted on the image, enabling intuitive and hands-free object identification.

The object detection component relies on the YOLOv8n model, a lightweight and efficient member of the YOLO family, known for its high-speed inference and competitive accuracy. As detailed in Algorithm 2, the detection process begins with importing necessary libraries such as OpenCV. Detection is performed via a single forward pass, producing bounding boxes, class indices, and confidence scores.

Each detected object is then processed to extract its bounding box coordinates (x_1, y_1, x_2, y_2), confidence score, and class label. These are used to annotate the image by drawing rectangles and overlaying label text. The annotated output is displayed to the user in real time using OpenCV's visualization functions. This end-to-end pipeline, combining speech-triggered commands and visual detection, offers an effective and lightweight solution for assistive technologies deployed on embedded platforms.

4.3. YOLOv8-Based Real-Time Object Detection

The image recognition module is responsible for identifying objects along with their two-dimensional positions within the image. YOLOv8, built upon a convolutional neural network (CNN) architecture, represents a major advancement in real-time object detection. Unlike traditional multi-stage approaches, YOLO formulates detection as a single regression problem by dividing the image into a grid and simultaneously predicting bounding boxes and class probabilities for each cell. This unified architecture enables high-speed and accurate inference, making it particularly suitable for embedded and real-time applications.

One key innovation in YOLOv8 is its streamlined architecture, incorporating Convolutional Block Attention Modules (CBAM), Feature Pyramid Networks (FPNs), and spatial attention mechanisms to improve detection accuracy across scales and challenging conditions. The network structure consists of three main parts:

- *Backbone:* A modified CSPDarknet53 responsible for initial feature extraction with residual and cross-stage connections.
- *Neck:* The C2f module and Spatial Pyramid Pooling Fast (SPPF) for multi-scale feature aggregation.
- *Head:* Up-sample and detection layers that output bounding boxes, objectness scores, and class probabilities at five scales.

This design balances performance and efficiency, making YOLOv8 ideal for embedded AI applications such as elderly care robotics.

4.4. Choice of Datasets

The choice of datasets is critical to the performance and robustness of the system. Two well-established datasets were selected:

- *Google Speech Commands Dataset:* The *Google Speech Commands Dataset* is a widely-used benchmark that provides a large collection of labeled one-second audio clips containing short voice commands such as "yes", "no", "stop", "go", and various object-related keywords. Developed by Google AI, the dataset is specifically designed to support the training and evaluation of lightweight voice recognition models, particularly for keyword spotting (KWS) tasks in embedded systems [30]. It includes over *105,000 audio samples* across 35 labeled classes, including directional commands (e.g., up, down, left, right), control words (on, off, stop, go), and special categories such as unknown (out-of-vocabulary inputs) and background noise (ambient or silent segments). Each file is a 16-bit PCM WAV file, sampled at 16kHz in mono format. The dataset features recordings from over *2,600 speakers*, capturing diverse accents, intonations, and acoustic scenarios. This diversity ensures strong generalization and robustness in real-world conditions. In our study, we utilized this dataset to train and test our speech recognition module in a *quiet indoor environment*. This setup allowed us to isolate the core recognition capabilities without external noise interference, ensuring reliable keyword detection performance during real-time operation. Given its well-structured vocabulary, breadth, and open availability, the Google Speech Commands Dataset remains a foundational resource for developing efficient speech-driven systems in embedded AI contexts, including assistive robotics.
- *COCO Dataset for Object Detection:* The Common Objects in Context (COCO) dataset is one of the most comprehensive and widely adopted benchmarks for object detection, segmentation, and captioning tasks. It contains over *330,000 images*, of which more than *200,000 are labeled*, spanning *80 object categories* such as bottle, cup, chair, and person—many of which are highly relevant for assistive and service robotics. The dataset includes over *1.5 million object instances* with pixel-level segmentation masks and contextual information in real-world environments. COCO's richness in scene complexity, occlusion, and object scaling makes it particularly valuable for training deep learning models aimed at robust, real-time object detection on embedded systems. In our work, it served as the foundation for pretraining object detection backbones, improving performance in cluttered indoor scenarios. The dataset is maintained by the Computer Vision group at Microsoft and is publicly available for academic use [31].

4.5. Model Selection: YOLOv8n

We employed the pre-trained YOLOv8n model provided by the Ultralytics framework. YOLOv8n (often called YOLOv8 Nano) is the lightest and fastest model in the YOLOv8 family—ideal for edge use cases with strict resource constraints—while offering respectable detection accuracy.

YOLOv8n was originally trained on the COCO dataset, which includes 200,000 labeled images across 80 object categories commonly encountered in real-world scenarios. This broad coverage enabled us to leverage the model without requiring additional training, making it an ideal choice for rapid prototyping and early-stage validation.

With its highly optimized architecture, YOLOv8n achieves a strong trade-off between detection accuracy and computational efficiency. Its small size (approximately 6.2 MB) and low inference latency make it well-suited for CPU execution during SIL evaluation, while also ensuring compatibility with Jetson Nano for future edge deployment.

Using the pre-trained model allowed us to focus on integrating the speech-based interface with visual recognition and to validate system functionality without the overhead of retraining.

4.6. Performance Metrics

To evaluate the effectiveness of the YOLOv8n model in object detection tasks such as traffic sign recognition, several key metrics are employed. These metrics provide a comprehensive understanding of the model's ability to make accurate predictions and to generalize well across varying conditions.

- *Accuracy is defined as the ratio of correctly predicted instances (true positives and true negatives) to the total number of instances.* It gives a general overview of the model's correctness:

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \quad (1)$$

where TP, TN, FP, and FN denote True Positives, True Negatives, False Positives, and False Negatives, respectively. Note that accuracy is useful in balanced datasets but can be misleading when one class dominates.

- *Precision* measures the proportion of true positive predictions among all instances predicted as positive:

$$Precision = \frac{TP}{TP+FP} \quad (2)$$

A high precision means that the model is reliable when it predicts a positive class, minimizing false positives.

- *Recall* (also known as sensitivity or true positive rate) evaluates the ability of the model to detect all actual positive instances:

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

High recall indicates that the model identifies most of the relevant items, reducing false negatives — essential in safety-critical domains like medical or assistive systems.

- *F1-Score* combines precision and recall into a single harmonic mean to balance both metrics:

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

The F1-score is most appropriate when both false positives and false negatives need to be minimized equally.

- *Mean Average Precision (mAP)* is a key metric for evaluating object detection models. It measures the average precision across all object classes:

$$mAP@0.5 = \frac{1}{N} \sum_{C=1}^{N} APc \tag{5}$$

where AP_c is the Average Precision for class c, typically computed from the area under the Precision-Recall curve:

$$APc = \sum_{n=1}^{N} (R_n - R_{n-1}).P_n \tag{6}$$

Here, P_n and R_n represent the precision and recall at the n^{th} threshold. A higher mAP score indicates both accurate classification and precise localization.

- *Precision-Recall (PR) Curves* plot precision against recall at various confidence thresholds. These curves help visualize the trade-off between precision and recall, aiding in threshold selection and model comparison, particularly when working with imbalanced datasets.

4.7. Training Output Structure and Performance Evaluation

The YOLOv8 model was trained and validated within a simulation environment on a PC equipped with an Intel i5 processor, NVIDIA GeForce RTX GPU, and 16GB RAM. The evaluation metrics include F1-score, precision, recall, and the precision-recall curve, providing insights into the model's performance under varying confidence thresholds. These metrics are critical to assessing the robustness and reliability of object detection prior to embedded deployment. As shown in Fig. 3, the F1-score varies with the confidence threshold applied to detection outputs. The overall F1-score curve, indicated in blue, reaches a peak value of 0.53 at a confidence threshold of 0.231. This result represents the best trade-off between precision and recall across all classes. Lower confidence thresholds permit more detections, including uncertain ones, which helps maintain recall at the expense of some precision. The relatively modest F1-score suggests that although the model is moderately effective, it may be affected by inference delays and reduced precision under SITL conditions.

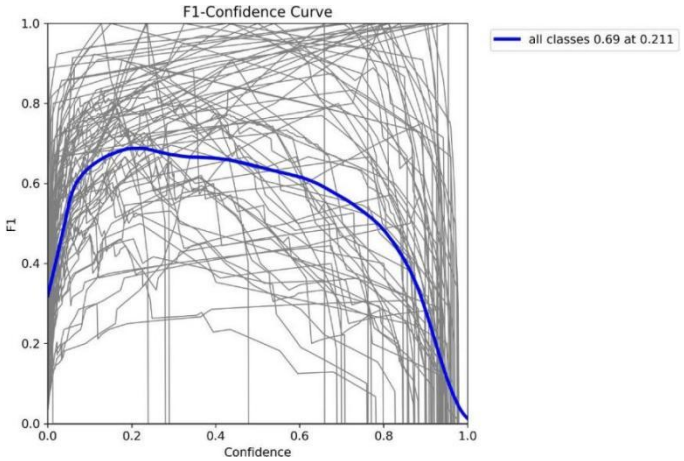

Fig.3. F1-Score vs. Confidence Threshold for YOLOv8

Fig. 4 illustrates the evolution of precision as a function of the confidence threshold. The global precision curve (in blue) steadily increases, reaching a maximum of 1.0 around a threshold of 0.974. This indicates that the model achieves perfect precision when it is highly confident, though it produces fewer predictions in such cases. While this behavior is desirable in critical applications, it also results in lower recall, highlighting the classic precision-recall trade-off and emphasizing the model's conservative behavior when operating under strict thresholds.

In Fig. 5, the precision-recall (PR) curve is plotted for all object classes. The blue curve represents the macro-average across all classes, while the gray lines indicate individual class performance. The model achieves a mean Average Precision at IoU threshold 0.5 (mAP@0.5) of 0.607, indicating strong detection performance across the dataset. This result confirms YOLOv8's strong performance on frequently occurring classes while showing a decline for less common or ambiguous categories. The curve further illustrates the decline in precision as recall increases, underlining the need to find an optimal balance point for deployment scenarios.

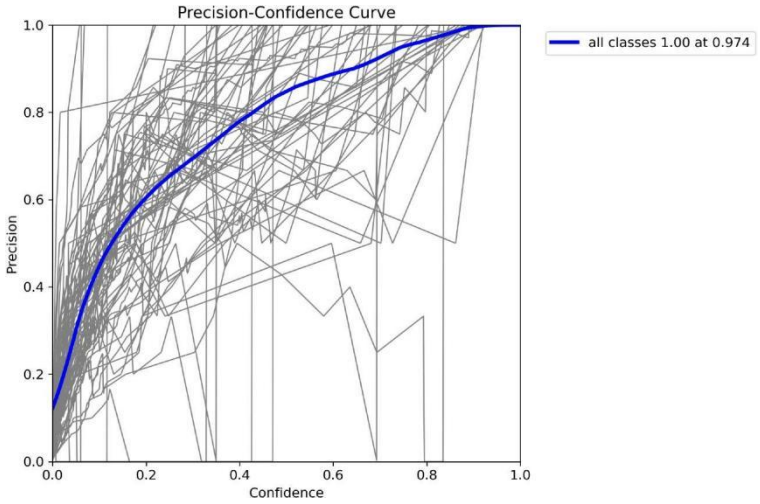

Fig.4. Precision vs. confidence threshold for YOLOv8

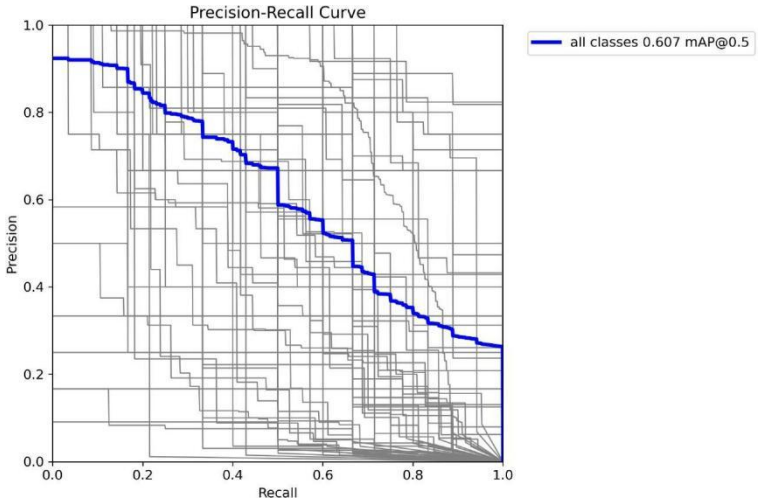

Fig.5. Precision-recall curve of YOLOv8

Finally, Fig. 6 presents the recall curve, highlighting the model's sensitivity to the confidence threshold. A maximum recall of 0.81 suggests that YOLOv8 can detect 81% of all objects when no filtering is applied. As the threshold increases, recall decreases—reflecting the filtering out of low-confidence yet valid detections.

The model maintains recall above 75% up to thresholds around 0.5, which is favorable for real-world applications requiring high object detection coverage. This performance further validates YOLOv8's robustness in the SITL environment prior to embedded deployment.

This stage of development allowed complete control over the software environment, enabling debugging and model evaluation before transitioning to the Jetson Nano in the HIL phase. The successful SIL validation confirmed the accuracy of both modules and ensured their compatibility, thereby setting the foundation for real-time deployment on embedded hardware.

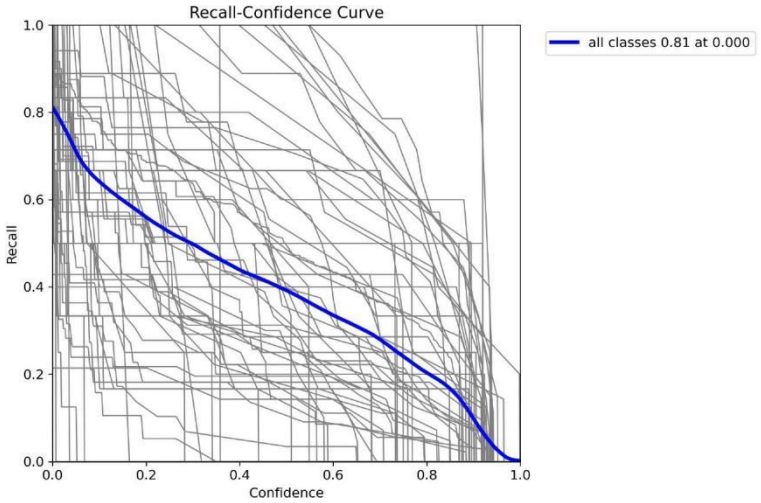

Fig.6. Recall vs. Confidence Threshold for YOLOv8

5. Model Optimization and Edge Deployment on Nvidia Jetson Nano

Following the successful software-in-the-loop development under Windows, the system was transitioned to the Jetson Nano platform for hardware deployment. This phase aimed to validate the end-to-end functionality of the system in realistic conditions, leveraging the Jetson Nano's embedded GPU and AI capabilities. The transition involved adapting both the speech recognition and object detection modules to run smoothly within the Linux for Tegra (L4T) environment.

5.1. Hardware Setup and System Integration

The deployment began with configuring and testing the USB microphone. We ensured the Jetson Nano correctly detected and could capture audio from the device. The speech transcription process was verified using a Python-based approach with an appropriate speech recognition library connected to an online API. Several challenges arose, including inconsistent device indexing and low input volume, which were mitigated through ambient noise calibration and manual configuration of the audio input source.

In parallel, the USB camera was tested using OpenCV to confirm it could stream live video reliably. A simple script was used to preview the feed and validate compatibility. Once operational, the YOLOv8 model was deployed and evaluated for real-time object detection. The model successfully identified and labeled objects in the environment, providing a functional visual interface for the system.

Integration of both modules followed. The user provides a verbal object command (e.g., "bottle"), which is recognized and transcribed into text. This keyword is passed to the YOLOv8 module, which scans each video frame for the corresponding object. If found, a bounding box and label are rendered in real-time, creating a complete speech-to-vision feedback loop.

Initially, the system was executed using only the Jetson Nano's CPU. This pipeline is illustrated in Fig. **??**, which outlines the sequential stages from speech recognition to final object detection. While this verified functional correctness, performance was limited to approximately 2–3 frames per second (FPS), resulting in noticeable latency. To address this, CUDA-based GPU acceleration was enabled.

5.2. Model Optimization via FP16 Precision and TensorRT

Deploying deep learning models on embedded platforms like the NVIDIA Jetson Nano requires meeting strict constraints in memory usage, computation speed, and energy efficiency. Among various model optimization techniques, including pruning [32] and knowledge distillation [33], reduced-precision inference has emerged as an effective strategy for accelerating inference while maintaining acceptable accuracy [34].

A. Precision Strategy for Embedded Inference

Reducing numerical precision allows deep learning models to achieve lower memory footprint, faster computation, and reduced energy consumption, which is particularly beneficial for edge deployment on resource-constrained platforms.

In this work, we focus on FP16 precision enabled by TensorRT on the Jetson Nano, which offers a favorable compromise between performance and ease of integration. Unlike Quantization-Aware Training (QAT), which requires retraining, FP16 precision does not alter model parameters and is natively supported by TensorRT and the Jetson GPU. Moreover, compared to integer quantization schemes such as INT8, FP16 precision avoids the accuracy degradation often observed for small or overlapping objects when calibration datasets are unavailable.

B. Deployment Pipeline on the NVIDIA Jetson GPU

For efficient inference on an NVIDIA Jetson device, the PyTorch-based YOLO model is first exported to the ONNX

format on a desktop machine. The ONNX model is then transferred to the Jetson Nano and converted into a TensorRT engine using NVIDIA's TensorRT toolkit. This two-step conversion—PyTorch to ONNX, followed by ONNX to TensorRT—enables hardware-specific optimizations and has proven to be an effective deployment strategy, as also confirmed by prior studies [35].

To deploy the model efficiently on Jetson Nano, we followed a multi-stage conversion process, summarized in Fig. 7:

- *YOLO Model Preparation*: The pretrained YOLOv8 model is obtained in PyTorch format.
- *TorchScript Conversion*: The model is optimized via torch.jit.trace() and torch.jit.optimize_for_inference().
- *ONNX Export*: The optimized model is exported to ONNX using torch.onnx.export().
- *TensorRT Compilation*: The ONNX model is parsed and compiled into a TensorRT engine configured for FP16 precision.

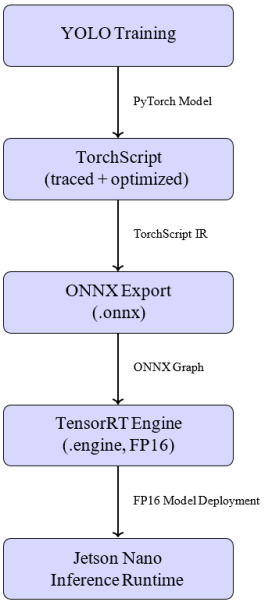

Fig.7. FP16 Precision and deployment pipeline: from YOLOv8 preparation to optimized inference on Jetson Nano's GPU

5.3. Performance Outcome

Following the implementation of the FP16-precision deployment pipeline, the object detection system was successfully deployed on the Jetson Nano, achieving significant improvements in execution speed and responsiveness. Initially, the YOLOv8 model was trained and validated on a desktop environment before being converted to the ONNX format and compiled into a TensorRT engine with FP16 precision. After installing GPU-accelerated versions of PyTorch and the Ultralytics framework, the pipeline was fully migrated to the Jetson Nano's embedded GPU.

This optimization enabled the system to reach over 950 frames per second (FPS) in synthetic benchmarks and consistently maintain 7–10 FPS during real-time video processing. Such performance is sufficient to support smooth interaction, confirming the suitability of FP16 precision for edge AI applications requiring low-latency inference, such as real-time object detection in assistive robotics for elderly care.

The edge deployment phase validated the robustness and practicality of the proposed system in real-world conditions. It demonstrated the feasibility of a voice-driven object detection assistant running entirely on embedded hardware, offering responsive, privacy-aware assistance in domestic environments tailored to elderly users. In the current prototype, assistance is provided through voice-guided object recognition and on-screen feedback rather than autonomous physical manipulation or task execution.

6. Edge Deployment Results

This section presents the experimental evaluation of the proposed voice-controlled object detection system deployed on the NVIDIA Jetson Nano platform. The goal is to assess its end-to-end functionality, processing speed, and detection accuracy under real-time embedded conditions. We compare CPU and GPU deployments and analyze system responsiveness based on inference speed and resource usage.

6.1. End-to-End System Validation

To validate the complete pipeline, the system was tested under real deployment conditions with integrated modules: speech recognition, object detection, and edge inference on Jetson Nano. As illustrated in Fig. 8, voice commands such as *"bring the bottle"* were accurately interpreted, triggering successful detection and localization of the target object in

the scene.

The output confirms the system's ability to handle multimodal input in real time, with consistent synchronization between audio processing and visual feedback. The visual confirmation of the detected object with bounding box overlay further validates the robustness of the pipeline. The real-time response, even under embedded hardware constraints, demonstrates that the deployed edge AI architecture meets the requirements for practical assistive robotics applications.

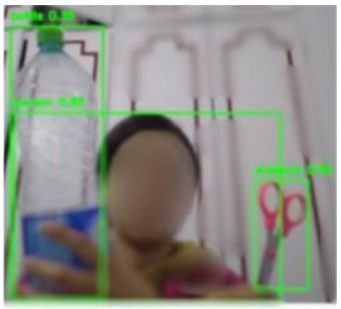

Fig.8. Output from live system: bottle detected upon voice command

6.2. Deployment on Jetson Nano's CPU

The voice-controlled object detection system was initially deployed on the CPU of the embedded Jetson Nano platform. As shown in Table 3, the system achieved an F1-score of 0.69 at a confidence threshold of 0.5. The average inference time per frame was approximately 130 ms, resulting in a frame rate of 7 FPS. While this level of performance is sufficient for offline evaluation and functional validation, it falls short of real-time processing requirements. The limitations are primarily due to the lack of GPU acceleration, which restricts the frame rate and increases latency, making it unsuitable for interactive or time-sensitive applications.

Table 3. Performance of the proposed voice-controlled object detection system on Jetson Nano's CPU

Metric	Value	Remarks
Inference Time	130 ms	CPU (PyTorch)
Frame Rate	7 FPS	Limited performance Satisfactory
F1-Score @ 0.5	0.69	detection

6.3. Deployment on Jetson Nano's GPU

To enhance performance, the proposed voice-controlled object detection system was optimized and deployed on the GPU of the Jetson Nano using TensorRT with FP16 quantization. As reported in Table 4, this configuration reduced the inference time to 70 ms per frame, nearly half the latency of the CPU version, and doubled the frame rate to 13 FPS, thus meeting real-time processing requirements. Importantly, the F1-score remained unchanged at 0.69, indicating that quantization preserved detection accuracy.

Table 4. Performance of the proposed voice-controlled object detection system on Jetson Nano GPU

Metric	Value	Remarks
Inference Time	70 ms	GPU (TensorRT + FP16)
Frame Rate	13 FPS	Real-time achieved
F1-Score @ 0.5	0.69	Post-quantization
Batch Size	8–16	VRAM optimized

FPS and latency measurements reflect the experimental conditions of the deployed embedded system and should be interpreted as indicative rather than as standardized benchmarks. Fig. 9 visually confirms the execution of the model on the Jetson Nano's GPU. The jtop interface shows that the GPU utilization reached 62.5%, while the memory usage allocated to the python3 process was approximately 82.6 MB. This demonstrates an efficient use of the embedded GPU, with TensorRT enabling lightweight yet powerful inference. The system maintains headroom for additional tasks or parallel processing, making it suitable for edge AI deployments under real-time constraints.

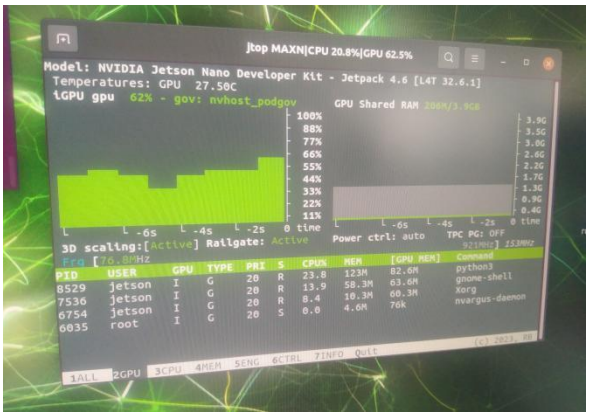

Fig.9. Inference running on Jetson Nano's GPU

These results demonstrate the feasibility of deploying deep learning models for object detection on low-power embedded platforms, without compromising performance or accuracy.

7. Conclusions and Future Work

This paper presented an edge-based AI framework for voice-controlled object recognition and generative interaction, targeting assistive robotic applications in elderly care. The proposed system integrates cloud-based speech recognition via the Google Web Speech API with real-time object detection using a pretrained YOLOv8n model accelerated through TensorRT with FP16 precision on the NVIDIA Jetson Nano. In addition, a locally hosted generative AI assistant provides conversational interaction without continuous Internet connectivity, contributing to privacy-aware on-device processing. Through careful model optimization and embedded deployment, the system achieved an F1-score of 0.69 for object detection while maintaining a real-time processing rate of 13 FPS and an inference latency below 70 ms. In our preliminary tests, the speech-to-text module achieved 94.3% command recognition accuracy; however, a fully reproducible evaluation protocol, including speaker diversity, sample size, noise-controlled conditions, and confidence intervals, will be provided as future work. Overall, these results demonstrate the feasibility of deploying a multimodal AI pipeline on resource-constrained embedded hardware, enabling responsive interaction in domestic environments. The proposed modular architecture supports low-latency perception and interaction while offering flexibility for future extensions in mobility, cognition, and decision-making. By decentralizing most AI components to the edge, the system reduces bandwidth requirements and limits continuous cloud dependency, thereby strengthening user trust through localized processing. Future research will focus on enhancing the autonomy, robustness, and adaptability of the proposed framework. While the current implementation already achieves efficient object detection on constrained hardware, further optimization will target lower power consumption and faster inference through advanced techniques such as integer quantization and structured pruning. To support real-world physical interaction, future developments will integrate motor control capabilities, enabling actions such as approaching or grasping objects in response to voice commands. Additional efforts will address robustness in noisy and dynamic environments by adopting fully local speech recognition models, incorporating microphone arrays for spatial filtering, and leveraging noise-augmented training data. The conversational component will also be extended toward multi-turn dialogue handling and task-level reasoning, allowing richer contextual interaction beyond single-step object localization. Enhancements to context retention and adaptive dialogue strategies are expected to improve personalization and usability, while hybrid neuro-symbolic reasoning approaches may be explored to increase transparency and safety in human-centric settings. Privacy considerations are currently limited by the use of a cloud-based speech-to-text service, which requires transmitting short audio segments to an external API. In future work, a dedicated privacy and threat analysis will be conducted, and cloud-based speech recognition will be replaced by an offline engine to enable end-to-end on-device processing. Although the system supports voice-guided object localization and interactive feedback, robotic pickup and delivery functionalities are not implemented in the current prototype and are considered future work. Finally, deployment in real elderly care environments will be pursued to assess usability, autonomy, and long-term assistive value through comprehensive user studies. Future work will include user studies and standardized dialogue evaluation metrics to quantify conversational coherence and user satisfaction. As future work, the object-detection component will be strengthened by defining an elderly-care-oriented subset of household object classes and by establishing a reproducible train, validation, and test protocol under representative home-environment conditions. The current prototype relies on the COCO-pretrained YOLOv8n model, which covers a broad set of everyday object categories. Performance metrics reported throughout the paper are indicative and correspond to representative operating points defined during experimentation.

All the Declarations and Statements

Author Contributions Statement
Sarra Ben Halima – Data curation, Implementation, Experiments, and Writing: Developed the system architecture, implemented the speech recognition and object detection modules, conducted experiments, and wrote the manuscript.
Faten Ben Abdallah – Conceptualization, Methodology, Supervision, and Review: Led the research direction, proposed the core ideas, designed the methodology, supervised all stages of the work, and critically reviewed and revised the manuscript.
Joseph Haggége – Supervision, Validation, Formal Analysis, and Review: Provided scientific supervision, contributed to the validation of the experimental results, supported the formal analysis, and participated in reviewing and editing the manuscript.

All authors have read and agreed to the published version of the manuscript.

Conflict of Interest Statement
The authors declare no conflicts of interest.

Funding Declaration
This research received no external funding.

Data Availability Statement
The datasets used in this study are publicly available.

Ethical Declarations
This study did not involve human participants or animal experiments.

Acknowledgments
We would like to express our sincere gratitude to the National Engineering School of Tunis and its faculty members for their administrative and technical support throughout this study. We also sincerely thank the members of the Automation Research Laboratory (LARA) for their valuable support and collaboration. We further thank the journal reviewers and editors for their insightful comments and suggestions, which improved the quality and clarity of this manuscript.

Declaration of Generative AI in Scholarly Writing
The authors declare that no generative AI or AI-assisted technologies were used in the preparation of this manuscript. All content, including text, figures, and analysis, was produced directly by the authors.

Abbreviations
The following abbreviations are used in this manuscript:

AI – Artificial Intelligence
API – Application Programming Interface
CNN – Convolutional Neural Network
COCO – Common Objects in Context
CPU – Central Processing Unit
CUDA – Compute Unified Device Architecture
DL – Deep Learning
FP16 – 16-bit Floating Point (Half Precision)
FPS – Frames Per Second
GPU – Graphics Processing Unit
HIL – Hardware-in-the-Loop
IoT – Internet of Things
KWS – Keyword Spotting
LLM – Large Language Model
mAP – Mean Average Precision
NLP – Natural Language Processing
ONNX – Open Neural Network Exchange
PR – Precision–Recall
QAT – Quantization-Aware Training
RAM – Random Access Memory
ReLU – Rectified Linear Unit

RMSE – Root Mean Squared Error
ROS – Robot Operating System
SAR – Socially Assistive Robotics
SIL – Software-in-the-Loop
SLAM – Simultaneous Localization and Mapping
TTS – Text-to-Speech
WHO – World Health Organization
YOLO – You Only Look Once

References

[1] World Health Organization. Mental Health of Older Adults. https://www.who.int/news-room/fact-sheets/detail/mental-health-of-older-adults, 2021.

[2] National Academies of Sciences, Engineering, and Medicine. *Social Isolation and Loneliness in Older Adults: Opportunities for the Health Care System*. The National Academies Press, Washington, DC, 2020.

[3] G. Peleka et al. RAMCIP—A service robot for MCI patients at home. In *Proc. IEEE/RSJ IROS*, Madrid, Spain, 2018.

[4] P. Asgharian, A. M. Panchoa, and F. Ferland. A Review on the Use of Mobile Service Robots in Elderly Care. *Robotics*, 11(6):127, Nov. 2022.

[5] S. Ben Halima, F. Ben Abdallah, and J. Haggége. Edge AI-based fall detection for an elderly care robot. In *Proc. IEEE 22nd Int. Conf. on Sciences and Techniques of Automatic Control and Computer Engineering (STA)*. IEEE, 2025.

[6] P. H. Varshini, V. Swati, D. Navya, and A. M. K. Aiswariya. EchoVision: Real-Time Object Detection and Voice Assistance for the Visually Impaired. In *Proc. 3rd Int. Conf. on Intelligent Systems, Advanced Computing and Communication (ISACC)*, pages 1–7, 2025.

[7] S. Cosar et al. ENRICHME: Perception and Interaction of an Assistive Robot for the Elderly at Home. *Int. J. Soc. Robot.*, 12:779–805, 2020.

[8] A. K. Pandey and R. Gelin. A mass-produced sociable humanoid robot: Pepper: The first machine of its kind. *IEEE Robot. Autom. Mag.*, 25:40–48, 2018.

[9] F. Fracasso et al. Social Robots Acceptance and Marketability in Italy and Germany: A Cross-National Study Focusing on Assisted Living. *Int. J. Soc. Robot.*, 14:1463–1480, 2022.

[10] F. Cavallo et al. Robotic services acceptance in smart environments with older adults: User satisfaction and acceptability study. *J. Med. Internet Res.*, 20(9):e264, 2018.

[11] G. A. Sai et al. Visio-Voice Transforming Images into Sound for the Visually Impaired. In *Proc. IEEE Int. Conf. on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS)*, pages 1–7, 2024.

[12] A. K. Luke et al. Envision: Assistance System for the Visually Impaired. In *Proc. 14th Int. Conf. on Computing Communication and Networking Technologies (ICCCNT)*, pages 1–8, 2023.

[13] R. K. Megalingam, P. T. K. Sai, A. Ashvin, P. N. Reddy, and B. R. Gamini. Trinetra App: A Companion for the Blind. In *Proc. IEEE Bombay Section Signature Conf. (IBSSC)*, pages 1–5, 2021.

[14] P. J. Duh et al. V-Eye: A Vision-Based Navigation System for the Visually Impaired. *IEEE Trans. Multimedia*, 23:1567–1580, 2020.

[15] G. Yang and J. Saniie. Sight-to-Sound Human-Machine Interface for Guiding and Navigating Visually Impaired People. *IEEE Access*, 8:185416–185428, 2020.

[16] R. C. Joshi, N. Singh, A. K. Sharma, R. Burget, and M. K. Dutta. AI-SenseVision: A Low-Cost Artificial- Intelligence-Based Robust and Real-Time Assistance for Visually Impaired People. *IEEE Transactions on Human- Machine Systems*, 2024.

[17] K. Jivrajani, S. K. Patel, C. Parmar, J. Surve, K. Ahmed, F. M. Bui, and F. A. Al-Zahrani. AIoT-Based Smart Stick for Visually Impaired Person. IEEE Trans. Instrum. Meas., vol. 72, pp. 1-11, 2023, doi: 10.1109/TIM.2022.3227988.

[18] G. Agarwal, K. Jindal, A. Chowdhury, V. K. Singh, and A. Pal. Image and Video Captioning for Apparels Using Deep Learning. *IEEE Access*, 2024.

[19] J. P. Docto, A. I. Labininay, and J. F. Villaverde. Third Eye Hand Glove Object Detection for Visually Impaired Using YOLO v4-Tiny Algorithm. In *Proc. IEEE Int. Conf. on Artificial Intelligence in Engineering and Technology (IICAIET)*, pages 1–6, 2022.

[20] D. Kleinberg, R. Yozevitch, I. Abekasis, Y. Israel, and E. Holdengreber. A Haptic Feedback System for Spatial Orientation in the Visually Impaired: A Comprehensive Approach. *IEEE Sensors Letters*, 7(9):1–4, 2023.

[21] M. Osama, A. Yehia, S. Mohamed, R. Sherief, N. Elmasry, V. Adel, and A. Hamdy. Design and Implementation of Visually Impaired Assistant System. In *Proc. Int. Mobile, Intelligent, and Ubiquitous Computing Conf. (MIUCC)*, pages 303–310, 2021.

[22] I. A. R. Djinko and T. Kacem. Video-based Object Detection Using Voice Recognition and YOLOv7. In Proc. 12th Int. Conf. on Intelligent Systems and Applications (INTELLI 2023), pp. 7-12, Barcelona, Spain, 2023.

[23] M. Gygli and V. Ferrari. Fast Object Class Labelling via Speech. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 15365–15374. IEEE, 2019.

[24] L. Ramesh, P. M. S. Pruthiga, M. Muthulakshmi, P. S. N. Sri, and S. Navaneetha. Voice Assistance for Visually Impaired Persons Using AI and IoT. In *Proc. 11th Int. Conf. on Bio Signals, Images, and Instrumentation (ICBSII)*. IEEE, 2025.

[25] R. Miyata, N. Yamaguchi, O. Fukuda, and H. Okumura. Object Search Using Edge-AI Based Mobile Robot. In *Proc. 6th Int. Conf. on Intelligent Informatics and Biomedical Sciences (ICIIBMS)*. IEEE, 2021.

[26] A. Gordeev, V. Klyachin, E. Kurbanov, and A. Driaba. Autonomous mobile robot with AI based on Jetson Nano. In *Proc. Future Technologies Conf. (FTC) 2020, Vol. 1*, pages 190–204. Springer, 2021.

[27] Anthony Zhang (Uberi). SpeechRecognition: Speech recognition module for Python. https://pypi.org/project/SpeechRecognition/, 2023. Accessed: 2025-07-17.

[28] Google Cloud. Cloud speech-to-text documentation. https://cloud.google.com/speech-to-text/ docs, 2025. Accessed: 2025-07-

17.
[29] Hubert Pham and PyAudio Contributors. PyAudio: Python bindings for PortAudio. https://pypi.org/ project/PyAudio/, 2023. Accessed: 2025-07-17.
[30] P. Warden. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *arXiv preprint arXiv:1804.03209*, 2018.
[31] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollar, and C. L. Zitnick. Microsoft COCO: Common Objects in Context. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 740–755. Springer, 2014.
[32] S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. *arXiv preprint arXiv:1510.00149*, 2015.
[33] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.
[34] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2704–2713, 2018.
[35] Y. Zhou and K. Yang. Exploring TensorRT to Improve Real-Time Inference for Deep Learning. In *Proc. IEEE 24th Int. Conf. on High Performance Computing & Communications (HPCC)*, 8th Int. Conf. on Data Science & Systems, 20th Int. Conf. on Smart City, and 8th Int. Conf. on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys), pages 2011–2018, 2022.
[36] I. Noreen, A. Akbar, and U. Siddiqui. AI-Enabled Elderly Care Robot. *J. Inf. Commun. Technol. Robot. Appl.*, 11(2):22–29, 2020.
[37] P. Kramomthong, C. Pintavirooj, and M. P. Paing. Smart Cane for Assisting Visually Impaired People and the Blind. In Proc. 13th Biomedical Engineering Int. Conf. (BMEiCON), pp. 1-5, 2021, doi: 10.1109/BMEiCON53485.2021.9745212.
[38] R. Ratheesh, S. R. Sri Rakshaga, A. Asan Fathima, S. Dhanusha, and A. K. Harini. AI-Based Smart Visual Assistance System for Navigation, Guidance, and Monitoring of Visually Impaired People. In Proc. 9th Int. Conf. on Science Technology Engineering and Mathematics (ICONSTEM), pp. 1-10, 2024, doi: 10.1109/ICONSTEM60960.2024.10568710., 2022.

Authors' Profiles

Ms. Sarra Ben Halima received the M.Sc. degree in automatic control and industrial computing from the Higher Institute of Applied Sciences and Technology of Kairouan, Tunisia, in 2023. She is currently pursuing a Ph.D. in Electrical Engineering at the National Engineering School of Tunis (ENIT), University of Tunis El Manar, and conducts her research at the Automation Research Laboratory (LARA) on modeling, control, and trajectory planning for autonomous and communicative mobile robots dedicated to elderly home assistance.

Since September 2025, she has been an Adjunct Lecturer at the Higher Institute of Computer Science (ISI), University of Tunis El Manar, Tunis, Tunisia. Her research interests include assistive mobile robotics for elderly home care, embedded and edge AI for real-time perception, human–robot interaction through speech and multimodal interfaces, safety monitoring in domestic environments, and learning-based navigation and motion planning under uncertainty.

Dr. Faten Ben Abdallah is currently an Assistant Professor at the National Engineering School of Tunis (ENIT), University of Tunis El Manar, Tunisia. She received the Engineering degree in electrical engineering and the M.Sc. degree in communication systems from ENIT. She obtained her Ph.D. degree in Signal Processing and Telecommunications from the University of Rennes I, France in 2009. She has more than 20 years of teaching experience in engineering schools at undergraduate and postgraduate levels.

She is a researcher at the Automation Research Laboratory (LARA), where she contributes to research on embedded and edge intelligence for healthcare engineering and assistive robotics, intelligent transportation systems and connected mobility, autonomous multi-platform robotic systems, and reconfigurable software-defined radio architectures. She has co-supervised Ph.D. and supervised Master's students and has served as a reviewer for several international journals and IEEE conferences. Her research outputs include peer-reviewed journal articles, Scopus-indexed publications, and book chapters, reflecting sustained contributions at the intersection of embedded intelligence, real-time systems, and autonomous robotics.

Prof. Joseph Haggége is a Full Professor of Electrical Engineering at the National Engineering School of Tunis (ENIT), where he teaches courses in electronics and automation. He heads ENIT's Automation Research Laboratory (LARA) and conducts research at the intersection of embedded systems, intelligent optimization, and artificial intelligence for control.

His research interests include metaheuristic optimization, hybrid and embedded systems, and robust digital control. His work combines methodological contributions in optimization-based decision and control with applied developments that account for real-time constraints on embedded platforms, with a focus on autonomous and cyber-physical systems.

Prof. Haggége has supervised and co-supervised Master's and PhD students and is actively involved in undergraduate and postgraduate engineering education. He has authored and co-authored peer-reviewed journal and conference publications and serves regularly as a reviewer in the areas of control, optimization, and embedded systems.

How to cite this paper

Sarra Ben Halima, Faten Ben Abdallah, Joseph Haggege, "Edge AI-Based Object Detection via Voice Recognition with an LLM-Based Emotional Assistant for Elderly Care Robots", International Journal of Information Technology and Computer Science(IJITCS), Vol.18, No.4, pp.50-67, 2026. DOI:10.5815/ijitcs.2026.04.04