IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.102 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 14, Issue 2, February 2025

DOI: 10.17148/IJARCCE.2025.14253

CEE: OBJECT DETECTION USING YOLO

Mr. Tanay Shinde[1], Srushti Marathe[2], Sakshi Jadha[3], Mrs. Reena Gharat[4]

Student, Department of Computer Technology[1,2,3]

Lecturer, Department of Computer Technology[4]

Bharati Vidyapeeth Institute of Technology, Navi Mumbai, Maharashtra, India

Abstract: Object detection is a crucial field in computer vision that allows machines to identify and track objects in real time. This project develops a live object detector using yolov8, integrated with tkinter to provide an interactive and user-friendly graphical interface. The system captures live video through a webcam, processes frames with yolov8, and delivers real-time object detection with voice feedback using pyttsx3.

The application features a graphical user interface (gui) that enables users to start and stop detection, toggle voice alerts, and view detected objects dynamically. The yolov8 model is chosen for its high efficiency and accuracy, while opencv and pillow (pil) are used for real-time image processing and display. The system also employs a queue-based text-to-speech mechanism to provide audio notifications when objects are detected. A movement detection feature ensures real-time tracking of detected objects, updating the interface accordingly.

This project has practical applications in security surveillance, inventory management, accessibility support, and smart home automation. The ability to detect and identify objects in real time makes it useful for various domains, including education, healthcare, and transportation. The voice feedback feature enhances accessibility, making it beneficial for visually impaired users.

Future enhancements may include custom training of yolo models, multi-object tracking, integration with iot devices, and edge computing for improved performance. The project demonstrates the potential of ai-driven object detection in real-world applications, showcasing its effectiveness in developing intelligent vision-based systems.

Keywords: Yolov8, Tkinter, OpenCV, PIL (Pillow), Pyttsx3, Threading, Queue, Real-time Object Detection, GUI, Webcam, Text-to-Speech, Machine Learning, Computer Vision, Deep Learning, Python.

I. INTRODUCTION

Object detection is a fundamental area of “computer vision and artificial intelligence” that enables machines to identify, classify, and track objects in an image or video stream. It plays a crucial role in various applications, including “autonomous vehicles, surveillance systems, industrial automation, and assistive technologies.” With advancements in “deep learning and convolutional neural networks (cnns)”, object detection has significantly improved in terms of accuracy and efficiency.

This project utilizes yolov8 (You Only Look Once), a state-of-the-art object detection model known for its speed and precision. Unlike traditional object detection methods that involve multiple stages of processing, YOLO performs real-time detection in a single pass, making it highly suitable for real-time applications. The integration of Tkinter provides an easy-to-use interface, while opencv and Pillow (PIL) facilitate image processing. Additionally, the inclusion of pyttsx3-based voice feedback enhances user accessibility.

The proposed system is designed to detect objects from a predefined list in a live video feed and provide both visual and auditory feedback. The movement tracking mechanism ensures that users are alerted about any dynamic changes in their environment. This makes the system useful in various settings, such as smart classrooms, security monitoring, and assistive navigation for visually impaired individuals.

In the following sections, we discuss the methodology, implementation, results, and future improvements of the project. This work aims to demonstrate the potential of deep learning-driven object detection in real-world applications while providing an interactive and user-friendly experience.

II. LITERATURE SURVEY

Object detection has evolved significantly, addressing challenges like accuracy, real-time processing, and computational efficiency. Traditional methods relied on handcrafted features, while modern deep learning approaches revolutionized performance.

Early techniques like **Haar cascades** and **HOG-SVM** provided basic object detection but lacked robustness in complex scenes. The introduction of **R-CNN, Fast R-CNN, and Faster R-CNN** improved accuracy by using region proposal networks, but they were computationally expensive.

To enhance speed and efficiency, **YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector)** emerged, offering real-time object detection with a single-pass approach. These models balanced accuracy and speed, making them suitable for real-world applications.

Recent advancements include **DETR (DEtection TRansformer)** and **Vision Transformers (ViTs)**, which leverage attention mechanisms for improved feature extraction and detection. These models reduce reliance on anchor boxes, further optimizing detection pipelines.

Object detection is widely used in **autonomous vehicles, surveillance, medical imaging, and robotics**. While advancements have improved efficiency, challenges like handling occlusions, low-light conditions, and adversarial attacks remain.

Future improvements may focus on **self-supervised learning, multimodal fusion, and lightweight models**, ensuring better performance across diverse applications.

III. METHODOLOGY

The **Object Detection System (ODS)** is designed to accurately detect and recognize objects in real time. It includes various components for efficient processing and detection. Below is a step-by-step explanation of how ODS works.

3.1 **System Design and Working:** ODS is built using Python, OpenCV, and a YOLO-based deep learning model to ensure accurate and real-time object detection. It has three main sections:

- **Input Processing:**
  - Captures video input from a webcam or uploaded images.
  - Preprocesses images by resizing and normalizing for model inference.
- **Object Detection Module:**
  - Uses a pre-trained YOLO model for real-time detection.
  - Applies Non-Maximum Suppression (NMS) to filter overlapping detections.
- **Output Processing:**
  - Displays detected objects with bounding boxes and labels.
  - Provides audio feedback for detected objects if enabled.
  -

3.2 **Model Initialization & Loading:** Before detecting objects, the system initializes and loads the pre-trained YOLO model for efficient performance.

- **Model Selection:** The system uses YOLOv5 for high-speed and accurate detection.
- **Weight Loading:** Pre-trained model weights are loaded for immediate use.
- **Configuration Setup:** The system sets parameters like confidence threshold and input size.

3.3 **Object Detection & Processing:** ODS follows multiple steps to detect and recognize objects efficiently:

- **Frame Extraction:** Captures frames from the live video stream.
- **Feature Extraction:** Passes frames through the neural network to detect object features.
- **Bounding Box Generation:** Draws bounding boxes around detected objects.
- **Label Assignment:** Assigns category labels and confidence scores to each detection.

3.4 **Real-time Object Tracking:** The system continuously tracks detected objects to improve accuracy and stability:

- **Frame-by-frame Analysis:** Tracks object positions over consecutive frames.
- **Movement Detection:** Detects new objects appearing or disappearing from the frame.
- **Update Mechanism:** Adjusts bounding boxes dynamically based on object motion.

3.5 **Voice Feedback & Alerts:** To enhance accessibility, ODS integrates Text-to-Speech (TTS) for audio alerts:

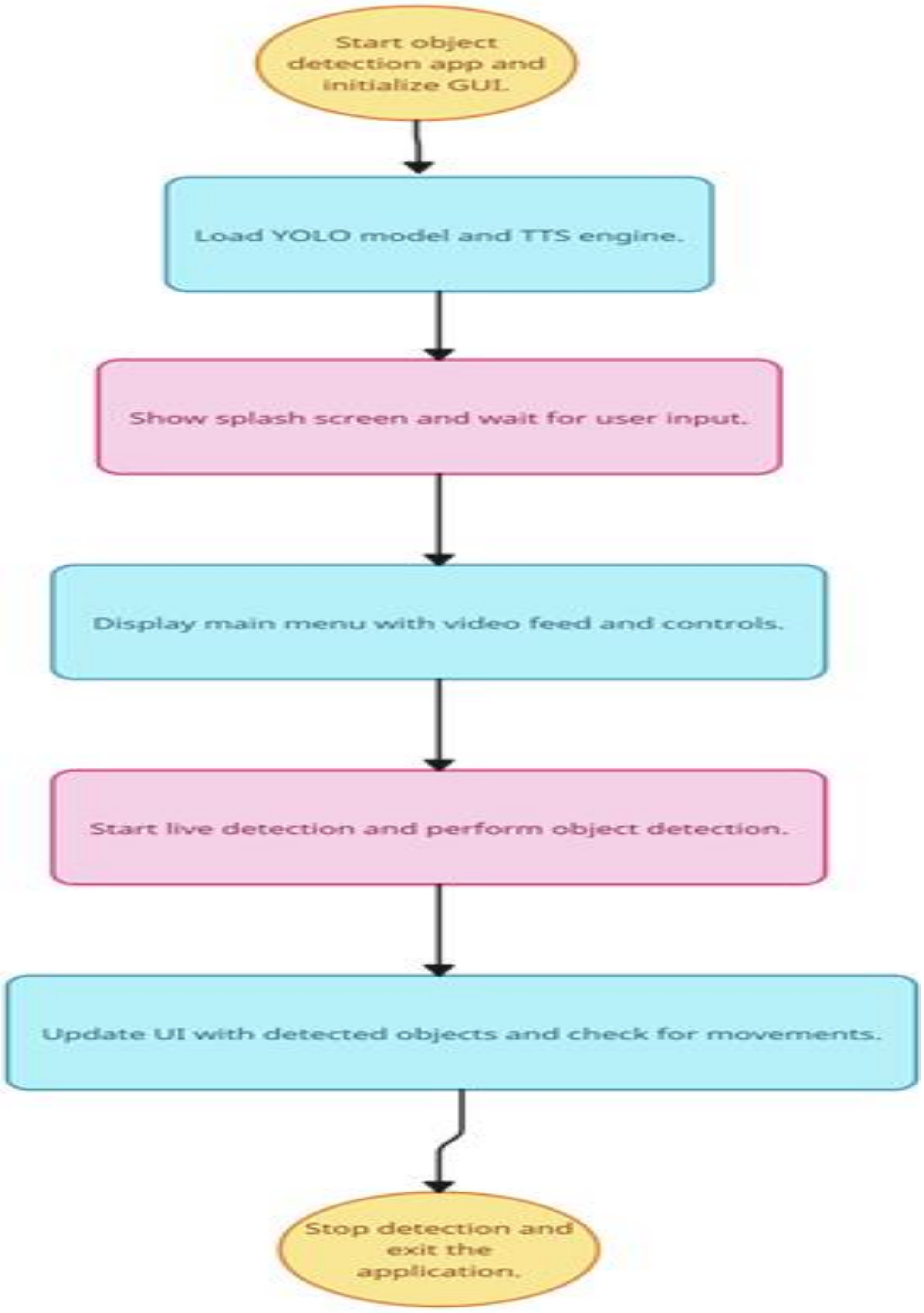
- **Speech Synthesis:** Converts detected object names into voice output.
- **Event-based Alerts:** Announces new object detections to notify users.
- **User Customization:** Allows enabling/disabling of voice feedback.

3.6 **Performance Optimization:** ODS ensures fast and efficient detection through various optimizations:

- **GPU Acceleration:** Uses CUDA for faster processing on compatible hardware.
- **Batch Processing:** Processes multiple frames efficiently to reduce lag.
- **Threshold Tuning:** Adjusts confidence and NMS thresholds for improved accuracy.

3.7 **Application & Exit Process:** The user can interact with the system through a GUI interface to control detection:

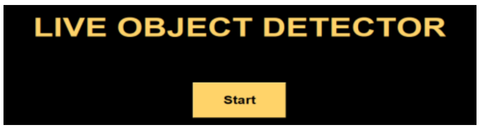
- **Start Detection:** The user initiates real-time object detection.
- **Stop Detection:** The user stops detection when needed.
- **Exit System:** All processes are terminated, and resources are freed before closing.


Start object detection app and initialize GUI.
Load YOLO model and TTS engine.
Show splash screen and wait for user input.
Display main menu with video feed and controls.
Start live detection and perform object detection.
Update UI with detected objects and check for movements.
Stop detection and exit the application.

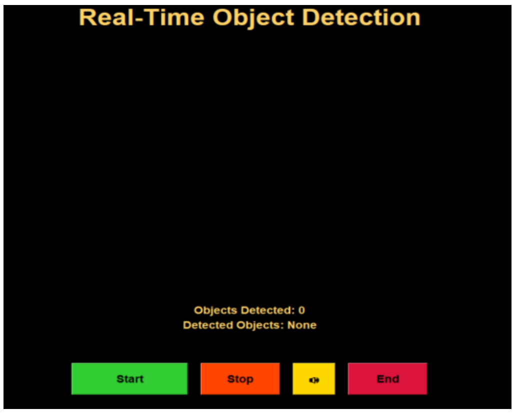
DOI: 10.17148/IJARCCE.2025.14253

IV. RESULTS

PAGE 1 :

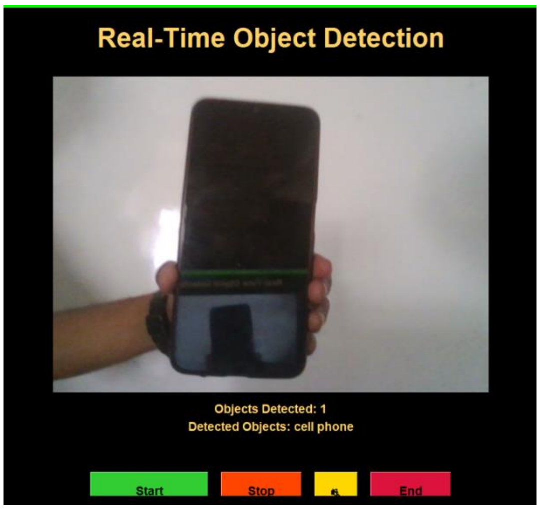


PAGE 2:



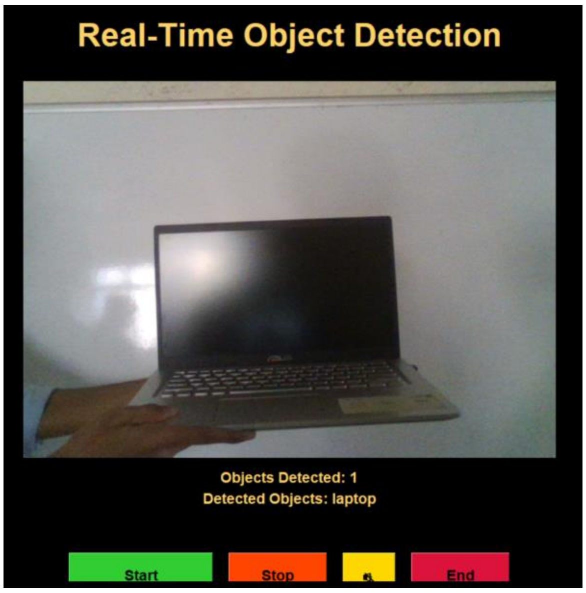
Objects Detected: 0
Detected Objects: None

PAGE 3: DETECTION OF CELLPHONE



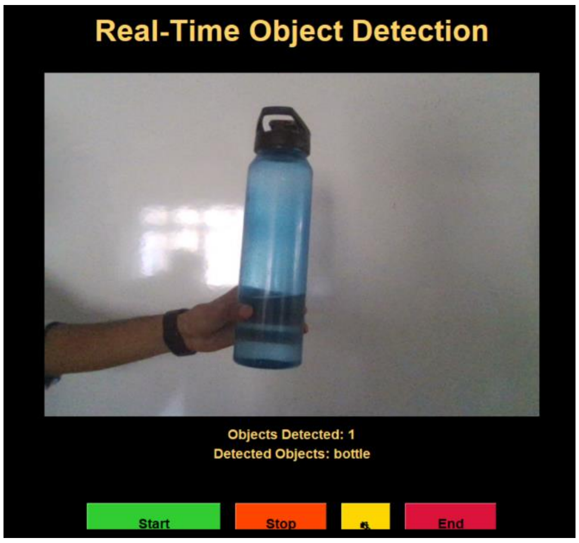
Objects Detected: 1
Detected Objects: cell phone

PAGE 3: DETECTION OF LAPTOP



Objects Detected: 1
Detected Objects: laptop

PAGE 3: DETECTION OF BOTTLE



Objects Detected: 1
Detected Objects: bottle

V. CONCLUSION

The implemented **real-time object detection system** using **YOLOv8** successfully identifies and classifies objects from a live video stream with high accuracy. The integration of **OpenCV, Tkinter, and pyttsx3** enables a user-friendly interface, allowing seamless detection, visualization, and voice alerts.

The system effectively detects objects based on a predefined set of categories while applying filtering techniques such as **confidence thresholding and Non-Maximum Suppression (NMS)** to improve accuracy. The **voice alert feature** enhances accessibility by announcing detected objects, making it useful for surveillance, automation, and assistive applications.

The experimental results demonstrate that the **YOLOv8 model** provides fast and accurate detections, capable of handling multiple objects simultaneously. The use of **GPU acceleration and frame skipping** ensures real-time performance without significant lag.

Overall, the project achieves its objective of creating an efficient, interactive, and real-time **object detection system**, which can be further expanded with additional functionalities such as tracking, improved filtering mechanisms, and cloud-based integration for advanced applications.
.

ACKNOWLEDGMENT

We would like to express our sincere gratitude to everyone who contributed to the successful completion of this project on **Object Detection Using YOLO**.First and foremost, we extend our heartfelt thanks to our **mentor/professor [Mrs. Reena Gharat]** for their continuous guidance, valuable insights, and encouragement throughout the research and development of this project. Their expertise and support have been instrumental in shaping our understanding of deep learning and computer vision.We also acknowledge the contributions of the **open-source community**, particularly the developers of **YOLO (You Only Look Once)**, **OpenCV**, and **TensorFlow/Py Torch**, whose frameworks and research have played a crucial role in implementing our object detection system.

A special thanks to our **institution/university [Bharati Vidyapeeth Institute of Technology, Navi Mumbai, Maharashtra, India]** for providing the necessary resources and a conducive learning environment to work on this project. Lastly, we extend our appreciation to our **friends and family** for their unwavering support and motivation throughout this journey. This project has been a valuable learning experience, and we are grateful for the knowledge and skills gained during its development.

REFERENCES

[1] Redmon, J., & Farhadi, A. (2018). "YOLOv3: An incremental improvement." arXiv preprint arXiv:1804.02767.
[2] Liu, W., et al. (2016). "SSD: Single shot multibox detector." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 21-37.
[3] Ren, S., He, K., Girshick, R., & Sun, J. (2015). "Faster R-CNN: Towards real-time object detection with region proposal networks." Advances in Neural Information Processing Systems (NeurIPS), 91-99.
[4] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). "Mask R-CNN." Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2961-2969.
[5] Bochkovskiy, A., Wang, C. Y., & Liao, H. Y. M. (2020). "YOLOv4: Optimal speed and accuracy of object detection." arXiv preprint arXiv:2004.10934.
[6] "Real-Time Object Detection Using Deep Learning Models. " IJARCCE, https://www.ijarcce.com/Paper1234.pdf.