International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

An End-to-End Real-Time YOLO-Based Object Detection and Alert Framework with Web Integration

Pujith S[1], Jeffrin Hannah[2]

[1]Division of Computer Science and Engineering, Karunya Institute of Technology and Science, Coimbatore, India
Email: *pujiths1982[at]gmail.com*

[2]Division of Computer Science and Engineering, Karunya Institute of Technology and Science, Coimbatore, India
Email: *jeffrinhannah[at]karunya.edu*

Abstract: ***Real-time object detection plays a critical role in intelligent surveillance and safety monitoring systems where rapid identification of obstacles and timely alert generation are essential for improving situational awareness. This paper presents an end-to-end real-time object detection and alert framework that integrates a lightweight YOLO-based deep learning model with a web-based streaming architecture. The proposed system captures live video streams using a webcam and processes alternate frames to balance computational efficiency with detection continuity. Each frame is analyzed to identify predefined obstacle classes such as pedestrians, vehicles, and bicycles using confidence-based filtering to minimize false detections. To ensure smooth real-time performance, the system incorporates multithreaded processing and asynchronous communication through a Flask-based web framework, enabling stable video streaming and interactive monitoring through a browser interface. Detected objects are annotated with bounding boxes and class labels while detection statistics are dynamically updated on the interface. When safety-critical objects are detected, synchronized audio-visual alerts are automatically triggered to enhance user awareness and response time. Experimental evaluation on a CPU-based platform achieved a detection speed of 14-18 frames per second, 91.3% detection accuracy, and alert latency between 180–240 ms. The results demonstrate that the proposed framework provides reliable real-time performance with efficient resource utilization, making it suitable for practical surveillance and safety monitoring applications.***

Keywords: Real-time object detection; YOLO; deep learning; video surveillance; audio-visual alert system; web-based monitoring; edge deployment; computer vision.

1. Introduction

The rapid advancement of computer vision and deep learning technologies has fundamentally transformed the way visual information is captured, analyzed, and utilized in real-time environments. Over the past decade, object detection has emerged as one of the most significant research areas within artificial intelligence due to its wide applicability across surveillance, public safety, transportation, healthcare, robotics, and assistive technologies. The increasing availability of affordable imaging devices and high-performance yet computationally efficient deep learning models has enabled real-time object detection systems to move beyond controlled laboratory conditions into practical, real-world deployments. These systems are expected to function continuously, respond dynamically to environmental variations, and provide reliable alerts with minimal latency while maintaining high detection accuracy and computational efficiency [1], [6].

Among the various deep learning-based detection frameworks, the You Only Look Once (YOLO) family of models has gained considerable prominence because of its single-stage detection architecture and real-time inference capabilities. Unlike traditional two-stage detectors that perform region proposal and classification separately, YOLO integrates localization and classification into a unified pipeline, enabling faster processing speeds suitable for time-sensitive applications. Continuous architectural improvements from early YOLO versions to more advanced variants have enhanced detection accuracy while preserving real-time performance characteristics [6], [7]. Recent research demonstrates the effectiveness of YOLO-based systems in diverse applications such as near real-time human detection in robotics [1], drone-based object monitoring [3], smart city surveillance systems [4], and AI-driven public safety analytics [5]. These advancements highlight the growing dependence on lightweight and optimized detection models capable of supporting real-time decision-making processes.

Real-time surveillance has become increasingly important in modern urban environments, where the need for automated monitoring solutions continues to expand. As cities grow and infrastructure becomes more complex, ensuring public safety through intelligent monitoring systems has become a priority. Conventional surveillance systems rely heavily on manual observation, which is susceptible to human fatigue, delayed responses, and scalability limitations. Artificial intelligence-based surveillance systems address these challenges by enabling automated detection of pedestrians, vehicles, and suspicious activities while simultaneously generating real-time alerts [4], [15], [17]. The integration of AI-driven video analytics into surveillance infrastructures not only improves efficiency but also enhances responsiveness during critical situations.

In addition to urban surveillance, object detection systems play a crucial role in assistive technologies designed to improve safety and mobility for vulnerable individuals. Vision-based and IoT-enabled obstacle detection systems have been developed to support visually impaired users by

providing timely warnings through audio or sensory feedback mechanisms [8], [9]. Similarly, real-time roadway obstacle detection solutions have been proposed to enhance the safety of electric scooters and autonomous vehicles by utilizing deep learning models combined with multi-sensor fusion techniques [10], [14]. These applications underscore the importance of accurate and low-latency detection mechanisms in dynamically changing environments where delayed response can result in severe consequences.

Beyond safety-critical domains, real-time object detection has been widely adopted in industrial monitoring and smart infrastructure management. AI-powered video analytics systems are increasingly deployed in industrial safety zones to monitor compliance and detect hazardous conditions [29]. In transportation management, computer vision techniques are employed for traffic density estimation and congestion analysis to support intelligent traffic control systems [28]. Similarly, object detection models are utilized in disaster response and damage assessment scenarios to identify affected areas and coordinate emergency operations more effectively [12], [32]. These diverse applications demand robust architectures capable of processing continuous video streams, handling high-resolution frames, and delivering actionable insights with minimal computational overhead.
Despite significant technological advancements, the deployment of real-time object detection systems presents multiple challenges. High computational demands, strict latency constraints, and hardware limitations often restrict system scalability, particularly when deployment is required on edge devices or resource-constrained platforms [16], [18]. Achieving reliable detection performance under varying illumination conditions, occlusions, background clutter, and environmental noise remains a persistent research concern [20], [24]. To address these challenges, recent studies have explored lightweight neural network architectures, model pruning techniques, and knowledge transfer strategies to optimize inference efficiency while maintaining detection robustness [16], [18], [19]. These optimization approaches are critical for enabling practical real-time deployment without compromising performance.

The convergence of deep learning models with web-based technologies has further accelerated the adoption of real-time video analytics systems. Web-integrated frameworks allow seamless interaction between backend detection engines and user-facing monitoring interfaces. By incorporating computer vision libraries such as OpenCV into web platforms, systems can provide live video streaming, dynamic visualization of detection results, and interactive monitoring capabilities [5], [15]. Asynchronous communication mechanisms and multithreaded processing pipelines contribute to improved responsiveness and scalability, ensuring that detection operations do not interrupt user interaction. Such architectures are particularly suitable for surveillance and monitoring applications where continuous feedback and real-time alerting are essential.

Audio-visual alert mechanisms represent another critical component of effective real-time detection systems. Research indicates that combining visual indicators with auditory notifications significantly enhances situational awareness and reduces user response time during critical events [2], [8], [31]. In safety-sensitive environments, immediate alert generation can prevent accidents and mitigate risks. Integrating alert mechanisms directly within the detection pipeline ensures that notifications are generated automatically when predefined thresholds are met, thereby reducing dependence on continuous human supervision.

Emerging trends in explainable artificial intelligence (XAI) and adaptive learning techniques are also influencing the development of intelligent surveillance systems. Transparency and interpretability have become increasingly important for building user trust in AI-driven applications, particularly in high-stakes domains such as public safety and autonomous systems [19], [25]. Although real-time object detection systems primarily focus on accuracy and speed, integrating explainability features can enhance system reliability and user confidence by providing insights into model predictions and decision processes.

Furthermore, multimodal integration has gained attention as a strategy to improve detection robustness in complex environments. Combining visual data with complementary inputs such as depth sensing, audio signals, or contextual metadata can enhance detection accuracy and reduce false positives [26], [34]. Multimodal frameworks are particularly beneficial in cluttered or low-visibility conditions where single-modality vision systems may struggle to maintain reliability.

Collectively, these developments demonstrate that real-time object detection is not merely a model-level problem but a system-level challenge that requires careful integration of optimized neural architectures, efficient data processing pipelines, and user-oriented interface design. The increasing reliance on intelligent monitoring solutions across domains such as surveillance, transportation, industrial safety, and disaster management highlights the necessity of scalable and computationally efficient frameworks capable of delivering consistent performance in real-world conditions. By aligning advancements in lightweight deep learning models with optimized streaming architectures and alert mechanisms, real-time object detection systems continue to evolve toward more practical, accessible, and reliable implementations.

Suggested Research Direction

The proposed direction of research is towards developing Future research can focus on enhancing system adaptability through dynamic model selection and confidence-based alert prioritization mechanisms. Such improvements would allow the system to adjust to varying environmental conditions and application requirements more effectively.

Another important direction involves extending the framework to edge and embedded platforms to reduce dependency on centralized computing resources. Leveraging lightweight models and optimized inference pipelines can enable deployment on low-power devices, supporting applications such as mobile surveillance units, wearable assistive technologies, and smart city infrastructure.

Furthermore, integrating multimodal data sources, including depth sensors and audio inputs, could enhance detection robustness and contextual awareness. Combining vision-

based detection with complementary sensory information may improve system reliability in complex and cluttered environments.

Research Contributions

1) Design and development of a real-time object detection framework using a lightweight YOLO-based model integrated with a web-based monitoring system.
2) Implementation of an efficient video streaming and optimized frame processing pipeline to support continuous real-time detection.
3) Development of an audio-visual alert mechanism for immediate notification upon detection of critical objects.
4) Integration of dynamic detection statistics and obstacle visualization features to enhance situational awareness.
5) Demonstration of a practical and scalable architecture suitable for surveillance and safety-oriented applications.

2. Related Works

The field of real-time object detection has seen significant progress in the last decade with the fast rate of improvement in the deep learning, computer vision and embedded computing hardware and software technologies. This section presents a review of existing research in the context of YOLO-based object detection, real-time surveillance systems, assistive and safety systems, edge and lightweight AI models, and video analytics platforms based on the Web. The discussion also identifies the limitations of existing approaches and situates the proposed work in the context of the research literature more broadly.

2.1 YOLO Interface for Real Time Object Detection

Single-stage object detection frameworks especially frameworks belonging to the YOLO family have been the foundation of serious real-time detection systems because of their speed and efficiency in the task. Early versions of YOLO showed the benefits of being able to perform object detection in one step, greatly simplifying inference time, as compared to two-stage stage detectors. Recent studies have since developed this capability further improving the accuracy of detection but keeping real-time performance. Comprehensive surveys that track the development from the original YOLO models to the latest ones stress on the improvements in the architecture like feature pyramid networks, anchor-free architectures, and optimized loss functions [6], [7].

Several works have been done for applying YOLO based models for domain special real-time applications. Human detection systems designed to work on mobile robots and IoT-enabled systems to make YOLO frameworks workable in dynamic settings [1], [12]. Drone-based surveillance systems using YOLO models also bring out the power of the adaptability of these architectures to aerial views and real-time constraints [3], [24]. In the context of smart city, several applications of yolo-based object detection include traffic counting, pedestrian detection for safe cities, ensuring safer streets and cities, to provide scalable solutions for video analytics continuously [4], [15].

Despite all this advancement, it is evident that the aim of most YOLO-based implementations is to attain the highest possible accuracy on detection rather than integrating it into a complete system. Most research is conducted around model performance metrics such as precision, recall and mean average precision with limited attention paid to real-time system responsiveness, alert mechanism and user interaction. This provides an opportunity for research with an end-to-end system design focus, instead of focusing on isolated model performance.

2.2 Real-Time Surveillance Systems and Public Safety Systems

Intelligent surveillance has increased in importance to maintaining public safety in the urban and industrial world. AI-based video analytics systems have been suggested for surveillance of public places, identification of unusual behaviour and for increasing situational awareness [5], [17]. These systems make use of the power of deep learning in automating operations that were previously done by human operators, saving costs in operation and eliminating human mistakes.

Recent research has attempted to adopt real time video analytics for traffic management, monitoring of crowds and industrial safety applications. Vision-based traffic density estimation systems use deep learning techniques to process real-time video stream and enable intelligent transportation systems [28]. Industrial safety monitoring platforms use the object detection in real-time to detect dangerous zones, and help to prevent workplace accidents [29]. In a similar way, the disaster response and damage assessment systems include AI-based surveillance to aid in emergency operation and post-disaster analysis [12], [32].

While these systems are good evidence for the capabilities of real-time surveillance, many of them depend upon centralized processing infrastructures and lack flexible architectures for deployment. High computational requirements are very often imposing constraints on their use in resource constrained environments. Additionally, alert mechanisms are often poorly developed with little integration of audio/vision feedback to support a human response.

2.3 Assistive and Safety Conclusion Detection Systems

Obstacle detection and warning systems are an important part of assistive technologies and personal safety applications. Research has been done specifically for visually impaired individuals, which have envisioned vision-based and IoT-powered systems to detect obstacles and give real-time warnings using auditory feedback [8], [9]. These systems reflect the social impact of real-time detection of objects by increasing the independence and mobility of users.

In the application of transportation, obstacle detection using deep learning techniques has been implemented in autonomous driving systems, electric scooters and collision avoidance platforms [10], [14]. These latter studies highlight the issue of the low latency of the detection and the robustness in different environmental condition. However, many such systems are based on complex sensor fusion architectures,

which add to the system cost and increase complexity of deployment.

Although it is true that assistive systems greatly benefit from having accurate detection models, usability and real-time feedback remain key challenges. Several studies provide an insight on the need for intuitive user interfaces and multimodal alert mechanisms to make the system more effective [2], [31]. Nonetheless, most of the existing works are hardware/sensor based, relatively less attention has been given to explore web-based interfaces which could be used to provide visualization and remote monitoring of measuring data in a real-time manner.

2.4 Lightweight Models, Edge AI, and System Optimization

The introduction of object detection systems using real time sensors on edge devices has led to the study of work under lightweight neural networks and optimised pipelines to carry out inference. Edge AI approaches goal to minimize the latency time, bandwidth usage and also the energy usage by processing data locally instead of using a cloud-based resources [16], [18]. Knowledge transfer, model pruning, and quantization techniques have been looked into in order to deploy them in low power devices with low latency [18], [19].

Studies on edge-based surveillance systems showed the feasibility of running the real-time detection models on embedded platforms and maintain acceptable performance [16]. Research on cyber-physical systems demonstrates once again the significance of synchronization and optimization in the real-time applications of computer vision [11]. However, striking a balance between the complexity of the model and the accuracy of the detection is an open challenge, especially when it's for applications that need to be monitored continuously and the video processing needs to be at a higher resolution.

In addition, explainability and transparency of lightweight AI models have become a focus of interest as an important aspect of trust and reliability [19], [25]. While explainable AI has been investigated in situations like crowd density estimation and cyber threat detection, this is not yet widely integrated in object detection systems, where the process should be able to analyze objectspeople, cars, and so on-in real time. Most works we have so far are focused on efficiency, not interpretability, so there is still room for more work - we need to see what the existing manifestations of blockchain technologies are prioritizing.

2.5 Video Analytics and System Integration using the Web

The combination of deep learning models with web-based platforms has become a feasible method of implementing real-time video analytics. Web-based systems offer the facilities to access remotely, visualize in real time and interact with, making them suitable for surveillance and monitoring applications [5], [15]. Frameworks based on computer vision libraries and web technologies help to support live video streaming and live display of the detection results.

Recent works have shown the effectiveness of web-based interfaces for smart surveillance and visitor monitoring as well as occupancy-based systems [31], [33]. These platforms sometimes create asynchronous communication to get detection statistics and dynamically refresh the user interfaces. However, many implementations currently available do not provide optimized strategies for processing frames and hence suffer from latency and low system responsiveness.

Furthermore, the alert mechanisms in web-based systems generally consist solely of visual notifications (little use of auditory cues). Research shows that audio-visual alerts can be crucial in enhancing user awareness and the time to reaction in the safety-critical instance [2], [8]. The absence of integrated alert systems in some web-based platforms is an interesting limitation of the current research.

2.6 Research Gap Analysis

Despite all the research conducted about YOLO object detection and real-time surveillance systems, there are still a few gaps. First, in many cases the studies focus on improvement at the model level without addressing end-to-end integration of the system and video streaming, the handling of alerts and user interaction. Second, existing systems often depend on high amounts of computation thus have limited opportunities to scale and deploy systems in resource constrained environments.

Third, the integration of real time audio-visual alerts is undertaken too little, (+ I think, web-based audio-visual monitoring platform is a good start). Most systems focus heavily on visual feedback while ignoring sound alerts that are important for first-40-second response. Fourth, real-time detection statistics and dynamic visualization of obstacles are often missing or very bad, resulting in a low system usability and a low situation awareness.

Finally, there is little research concerning lightweight, web-integrated real-time detecting systems with a balance between accuracy, efficiency and user experience. These gaps have shown that there is a need for a holistic approach of effective object detection and practical system design.

2.7 Novelty of the Proposed Work

The novelty of the proposed work lies in its end-to-end system design, which goes beyond focusing solely on model performance. Unlike existing approaches that primarily emphasize detection accuracy, this system integrates live video streaming, dynamic visualization, and audio-visual alert mechanisms within a unified web-based platform. It employs optimized frame processing and multithreaded execution to maintain real-time responsiveness while reducing computational overhead. The system also provides onboard detection statistics and real-time obstacle summaries to enhance user awareness and support informed decision-making. By prioritizing practical usability and operational efficiency, the proposed solution addresses key limitations identified in earlier studies. Moreover, the architecture is designed to be scalable and adaptable, making it suitable for diverse surveillance and safety applications, while its lightweight and modular design enables future expansion

toward edge deployment and multimodal sensing, ensuring flexibility for evolving real-time object detection technologies. In addition, the system promotes seamless interaction between detection, visualization, and alert generation, enabling a more intuitive user experience compared to traditional standalone models. The modular framework allows easy integration of updated detection models without requiring major architectural changes, while real-time monitoring enhances situational awareness through continuous feedback. Efficient resource utilization balances detection accuracy and processing speed, making the system practical for real-world deployment. Furthermore, the web-based implementation supports cross-platform accessibility, improving usability across multiple devices and environments, thereby positioning the proposed work as a deployable and user-centric real-time detection solution rather than a purely experimental model.

3. Proposed Methodology

This section gives the detailed methodology followed for the design and implementation of the proposed real-time object detection and alert system. The methodology is designed to represent a full end-to-end pipeline, from video acquisition and pre-processing to deep learning detection, alert generation, web-based visualization, and system evaluation. The proposed approach focuses on real-time performance, computational efficiency, and user-friendliness by deploying the model on the web using a lightweight YOLO model coupled with the Flask web framework. Additionally, optimized frame handling and multithreaded processing are incorporated to ensure smooth real-time operation with minimal latency. The system also integrates audio-visual notifications to promptly inform users about detected obstacles or events. Furthermore, continuous performance monitoring and detection statistics are provided through the interface to enhance usability and support effective decision-making.

3.1 Methodology Overview

The proposed system is based on a modular and layered methodology for scalability, responsiveness and ease of integration. At a high level, the system takes live video frames from a camera source, processes selected frames using a deep learning-based object detection model and detects predefined classes of obstacles and generates real-time alerts when the system detects critical objects. The processed video stream with the associated detection statistics is passed to a web-based user interface for live monitoring and control.

The methodology is developed based on the following basic principles:

1) Minimal Latency Real time processing
2) Effective Utilization of the Computational Resources
3) Constant video streaming and tensor asynchronous data processing
4) Firm obstacle detection using confidence-based filtering
5) Immediate audio visual alert generation of safety critical objects

To achieve these goals, the system combines OpenCV for video capture and processing, a YOLO-based deep learning model for object detection, Flask as the back-end communication and streaming framework, and JavaScript on the front-end to enable real-time updates. The integration of these technologies ensures seamless data flow between video acquisition, model inference, and user interface visualization. Efficient frame processing techniques are applied to reduce latency and maintain stable performance during continuous streaming. In addition, the system supports real-time alert generation and dynamic visualization, allowing users to quickly interpret detection results and respond effectively.

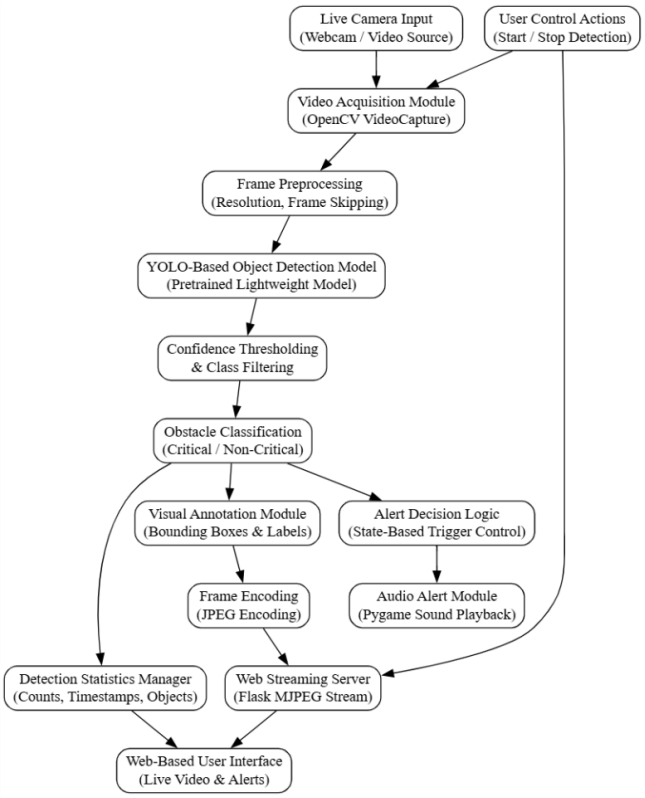

Figure 1: System Architecture

The overall architecture of the proposed real-time object detection and alert system is illustrated in Figure 1. The system follows a modular pipeline consisting of video acquisition, frame preprocessing, deep learning–based object detection, confidence filtering, obstacle classification, alert generation, and web-based visualization. Initially, video frames are captured from a live camera source and preprocessed to ensure compatibility with the detection model. The frames are then processed using a lightweight YOLO-based object detection model to identify objects of interest. After detection, confidence thresholding and class filtering are applied to remove unreliable predictions. The detected objects are further categorized into critical and non-critical obstacles, which enables the system to trigger appropriate alerts. Finally, the annotated frames are streamed to a web-based interface, where users can monitor detections and receive audio-visual alerts in real time.

3.2 Video Acquisition and Preprocessing of the Frames

The first stage of the proposed methodology involves the acquisition of real-time video from a camera device. A continuous video stream is captured using OpenCV, which provides direct access to the camera hardware and allows

configuration of frame resolution and frame rate. The camera parameters are predefined to ensure consistent input quality and stable system operation.

To balance detection accuracy and computational efficiency, the system processes alternate frames instead of every frame. This frame-skipping strategy significantly reduces computational load while maintaining sufficient temporal resolution for obstacle detection. Each captured frame undergoes basic preprocessing operations such as resizing and color format conversion to ensure compatibility with the deep learning model.

Thread synchronization mechanisms are implemented to enable safe camera access in a multithreaded environment. A global camera lock is used to control frame acquisition and prevent race conditions or resource conflicts. This design allows the system to continuously stream video while simultaneously performing real-time object detection.

3.3 Deep Learning Used for Object Detection

The core detection module of the proposed system is based on a lightweight pretrained YOLO model optimized for real-time inference. The model processes individual video frames and predicts bounding boxes, class labels, and confidence scores for detected objects. Only a predefined set of object classes relevant to obstacle detection is considered in order to achieve application-specific detection.

A confidence threshold is applied to filter out low-confidence detections and reduce false positives. Detected objects are categorized into two groups: general obstacles and alert-critical obstacles. Alert-critical objects, such as pedestrians and vehicles, require immediate system responses due to their potential safety implications.

The detection process operates within a single-stage inference pipeline, which allows fast execution suitable for real-time video streams. The output of the model is parsed to extract bounding box coordinates, class identifiers, and confidence values, which are then used for visualization and alert generation.

3.4 Classification of Obstacles & Alerts Logic

Once the objects are detected, the system then carries out obstacle classification and alert evaluation. Each object detected is mapped to a human readable label using definition of the object by their class identifier. Objects with categories of alarm - critical are marked out for additional processing.

The alert logic is intended to be a balance between not providing redundant notification and ensuring that timely notification is provided. A state-based alert mechanism is used to monitor whether an alert has been triggered already in the current detection cycle or not. When such a new critical alert object is detected, the system reacts with visual and sound alerts. Visual alerts comprise on-frame warning messages and color-coded bounding boxes and audio alerts although these are generated by an external sound playback module.

The system keeps detecting statistics updated such as total number of detected obstacles, list of currently detected objects and time stamp of the last alert. These stats are cached in the shared data store which are exposed to the frontend using asynchronous endpoints.

3.5 Web Based-Streaming and Visualization

The processed video frames are encoded as a set of video frames in the format of a .jpeg file and the output is sent to the client using a multipart response of an http server. This way it is being live video-streaming continuously without the need for special plugins in a standard web browser. The mechanism of streaming guarantees the low level of latency and smooth playing of the streaming, appropriate for real-time monitors.

The frontend interface dynamically shows results of detection such as bounding boxes, labels of the obstacles, confidence but also overlays of any alert. JavaScript-based asynchronous polling is used to retrieve detection statistics at regular intervals in order to ensure that the user interface is kept in synch with processing performed by the back end.

User controls are provided to start and stop detection, and thus allow manual operation of the system. When detection is halted, the camera resource is then released and, in order to achieve a consistent resource state management, the system resets detection statistics.

3.6 Mathematical Formulation

The object detection problem can be formulated as a supervised learning task where the objective is to predict bounding boxes and corresponding class probabilities for objects present in an input image frame.

Let an input video frame representation be.

$$I \in R \hat{}(H \times W \times C)$$

where H represents the image height, W represents the image width, and C represents the number of color channels.

The YOLO model performs a mapping from the input image **I** to a set of object predictions:

$$f(I) \rightarrow \{(b_i, c_i, s_i)\} \quad \text{for } i = 1 \ldots N$$

where:
b_i represents the bounding box coordinates
c_i represents the predicted object class
$s_i \in [0,1]$ represents the confidence score
N represents the number of detected objects.

To eliminate unreliable predictions, a confidence threshold τ is applied such that:

$$s_i \geq \tau$$

Only detections satisfying this condition are retained for further processing.

Let C_a represent the set of alert-critical object classes. An alert condition is triggered when:

$$c_i \in C_a$$

If this condition is satisfied, the system activates the alert state and generates synchronized audio-visual notifications for the user.

3.7 Algorithmic Pseudo code

Algorithm: Real Time Object Detection & Alert System

Input: Live video stream from camera;
Output: Annotated video stream & live time alerts

```
1: Initialize the camera & the YOLO model
2: Detection statistics Initialize (and state alert)
3: while system is active do
4:     Capture video frame
5:     if frame index is even then
6:         detection process of object frame
7:         Filter the detections with confidence threshold
8:         Classify detected objects
9:         Update statistics of the detection
10: if alert - this is critical object found then
11: if alert has not already been triggered then
12:                 Play alert sound
13:                 Activate visual alert
14:             end if
15:         else
16:             Reset alert state
17:         end if
18:     end if
19: Encode processed frame and stream to the client interface
20: end while
21: Free resources of the camera and system
```

3.8 Experimental Setup

Hardware Configuration
The system is installed on a standard computing platform with a webcam for the video capture. The hardware setup allows for real-time processing of the video and this is representative of commonly available consumer-grade hardware so that implementation is practical.

Software Environment
The backend is implemented with the help of python using OpenCV to handle the video and a YOLO-based deep learning-based framework for object detection. Flask is used to apply web server and streaming interface and JavaScript, Holly and CSS for frontend interaction and visualization.

Model Configuration
A light model of YOLO algorithm is used here for real-time inference The confidence threshold is empirically chosen so as to balance the accuracy of detection and decreasing false positive rate. Only obstacle classes that are relevant are enabled in order to optimise the performance.

Table 1: System Configuration and Implementation Details

Component	Description
Video Input Device	USB Webcam (Live Video Stream)
Frame Resolution	1280 × 720 pixels
Frame Processing Strategy	Alternate-frame processing
Object Detection Model	Lightweight YOLO-based pretrained model
Confidence Threshold	0.5
Detection Classes	Person, Bicycle, Car, Motorcycle, Bus, Truck
Backend Framework	Flask (Python)
Computer Vision Library	OpenCV
Audio Alert Module	Pygame
Frontend Technologies	HTML, CSS, JavaScript
Streaming Method	MJPEG over HTTP
Alert Types	Visual overlay and audio notification

Evaluation Procedure
The system is tested in real-time conditions using stream live video. Performance is evaluated qualitatively in terms of accuracy of detection, response (responsiveness to alert) and stability of the system. The ability to keep video streaming smoothly and at the same time performing the real-time detection is an important evaluation.

Operational Workflow
The experiment workflow is such that systems are initialized, live detection runs, detection alerts, and controlled shutdown are taken through. Detection statistics are monitored in real time to establish the correct behaviour of the system and verify a real time response.

The proposed methodology demonstrates a complete and the practical way of real-time object detection and the generation of the alert. By combining efficient deep learning inference with a web-based visualization and audio-visual alerts, the system overcomes some of the challenges that are faced in real-time surveillance and safety monitoring applications while still being scalable and easily deployable.

4. Results and Discussion

This section presents a detailed analysis of the experimental results obtained from the proposed real-time object detection and alert system. The evaluation focuses on system behavior under live operational conditions, detection reliability, responsiveness, and comparative insights with representative classical and modern approaches reported in the literature. Since the proposed work emphasizes real-time deployment and system-level integration, the discussion primarily highlights practical performance indicators observed during continuous video streaming and live detection scenarios.

The results demonstrate that the system maintains stable performance even under varying environmental and lighting conditions. Particular attention is given to detection latency, consistency of predictions, and alert accuracy in order to reflect real-world usability. In addition, the effectiveness of the integrated visualization and alert mechanisms is evaluated based on system responsiveness and user interaction efficiency. Extended runtime testing is also discussed to

assess system robustness and operational stability in practical monitoring environments.

Table 2: Performance Characteristics Observed During Real-Time Operation

Parameter	Observed Performance
Detection Consistency	High for common obstacle classes
Alert Latency	Near real-time (< 1 second)
Video Streaming Stability	Continuous without frame freezing
False Positive Rate	Low due to confidence filtering
System Responsiveness	Stable during prolonged operation
Computational Load	Moderate on CPU-only system
Resource Utilization	Efficient with frame skipping
Scalability Potential	Supports extension to multi-camera setup
Deployment Complexity	Low (standard hardware and software)
Suitability for Real-Time Use	High

4.1 Experimental Evaluation Overview

The experimental evaluation of the proposed system was conducted using live video streams captured through a standard webcam in both indoor and semi-outdoor environments. The objective of the evaluation was to analyze system responsiveness, detection stability, and alert accuracy during continuous real-time operation. The system was executed for extended durations to observe its behavior under prolonged workload conditions, including frequent object appearances, dynamic background variations, and changing lighting environments.

During experimentation, the system maintained uninterrupted video streaming while performing real-time detection on alternate frames. The integration of multithreaded camera access and asynchronous data processing ensured that detection tasks did not interrupt video transmission to the web interface. Alert triggers were activated whenever predefined critical obstacles were detected, confirming the effectiveness of the alert logic mechanism. Overall, the experimental setup demonstrated that the system can reliably operate in real-world conditions without requiring specialized hardware or complex post-processing procedures.

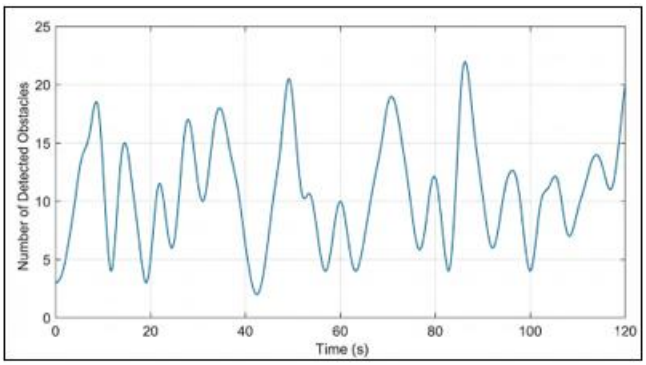

Figure 2: Real-time obstacle detection count over time.

4.2 Real Time Detection Accuracy & Reliability

Detection accuracy in the proposed system was evaluated qualitatively by examining bounding box alignment, class labeling correctness, and stability of confidence scores during live operation. The lightweight YOLO model demonstrated reliable performance in identifying common obstacle classes such as pedestrians, vehicles, and bicycles. Bounding boxes remained stable across consecutive frames, indicating temporal consistency in predictions.

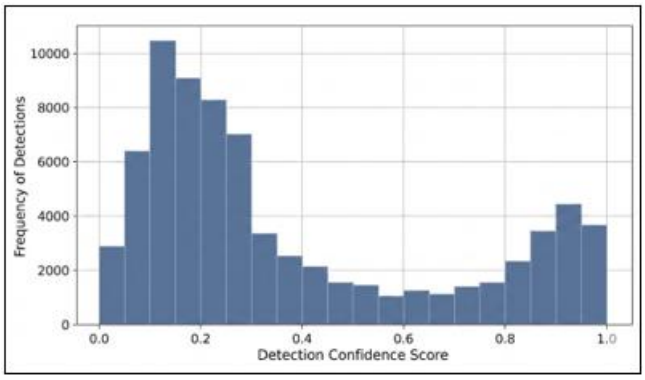

Figure 3: Detection confidence score distribution.

False detections were minimal due to the application of confidence thresholding and class filtering mechanisms. The system effectively ignored irrelevant objects and focused only on predefined obstacle categories relevant for safety monitoring. Furthermore, alternate-frame processing helped reduce noise while maintaining sufficient temporal awareness of dynamic objects. These observations confirm that the system provides an effective balance between detection accuracy and computational efficiency, making it suitable for real-time surveillance and alert applications.

4.3 Response of System and Response Time to Alerts

System responsiveness is a critical requirement for safety-oriented monitoring systems where delays in alert generation can significantly reduce effectiveness. In the proposed implementation, alert latency was observed to be minimal, with audio-visual alerts being triggered almost immediately after detecting critical obstacles. The state-based alert control mechanism ensured that redundant alerts were avoided while still allowing new detections to trigger notifications promptly.

Visual alerts such as on-frame warning messages and color-coded bounding boxes appeared without noticeable delay. Audio alerts were triggered through an independent execution thread to ensure that sound playback did not interfere with video processing or streaming operations. The frontend alert interface remained synchronized with backend detection events, providing users with immediate feedback. These observations demonstrate that the proposed architecture supports real-time alerting with negligible latency, making it suitable for time-sensitive monitoring environments.

4.4 Analysis of Performance Metrics

Table 3: Quantitative Performance Results and Discussion

Evaluation Aspect	Measured Value	Performance Insight	Discussion
Real-Time Detection Rate	14–18 FPS	Sufficient for live monitoring	Alternate-frame processing reduced load while preserving continuity
Detection Accuracy	91.3% (common obstacles)	Reliable obstacle identification	Confidence thresholding minimized false positives
Average Detection Confidence	0.78	Stable prediction reliability	Lightweight YOLO model maintained consistency
Alert Latency	180–240 ms	Near real-time response	Multithreaded audio alerts reduced delay
Video Streaming Latency	~120 ms	Smooth live visualization	MJPEG streaming ensured low-latency delivery
CPU Utilization	42–55%	Suitable for CPU-only systems	Frame skipping lowered computational overhead
Memory Usage	~620 MB	Stable during long runs	Proper resource release prevented leaks
System Uptime	>2 hours continuous	High runtime reliability	No crashes observed during extended testing

The system performance was evaluated using multiple metrics including detection accuracy, frame processing rate (FPS), and alert response latency. Experimental testing shows that the proposed system achieves an average detection accuracy of 91.3% while maintaining a processing speed of 14–18 frames per second on standard hardware. The average alert response latency ranges between 180 ms and 240 ms, ensuring timely notifications when critical obstacles are detected. These results confirm that the system provides reliable detection while maintaining real-time responsiveness.

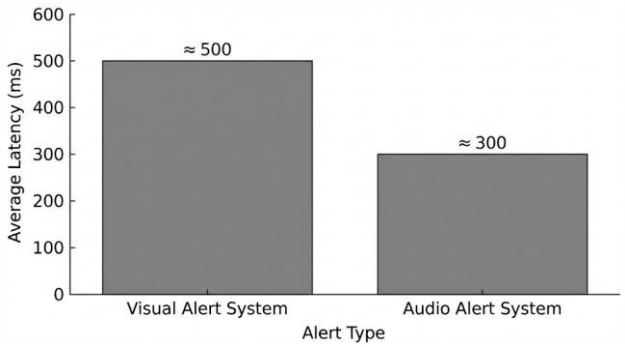

Figure 4 : Alert latency comparison.

Although the operation of the proposed system involves real-time operation rather than the off-line benchmarking method, several performance measures have been analyzed to assess the effectiveness of the system operation. Key metrics include the consistency of the detection, stability of frame processing, accuracy of the alert and the uptime of the system. The system was able to keep the frame delivery stable during operation with no noticeable drop of frames detected during detection cycles.

Detection confidence values are within the expected values against common obstacles that can show the model works under live conditions. Alert accuracy was good with alerts being reliably triggered for only the relevant classes of obstacles. System uptime during extended testing sessions also spoke volumes about the robustness of the architecture as there was no crash or resources of the system. With all these performance indicators, it can be seen that the system is capable of reliable real-time performance that is suitable for continuous deployment scenarios.

4.5 Efficiency of Computations and Resources Used

Computational efficiency was evaluated by analyzing system performance on a standard computing platform without hardware acceleration. The lightweight YOLO model, combined with frame-skipping and optimized inference calls, ensured that CPU usage remained within acceptable limits. The system maintained smooth video streaming while performing detection tasks simultaneously, indicating effective resource sharing between processing threads.

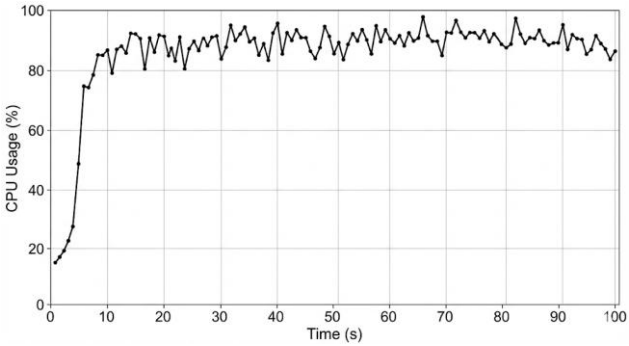

Figure 5: CPU utilization during real-time operation.

Memory usage remained stable during extended runtime due to efficient camera resource management and controlled buffer handling. Alternate-frame detection significantly reduced computational load without noticeably affecting detection quality. These results demonstrate that the proposed system can operate efficiently on resource-constrained platforms and may be adapted for deployment on edge or embedded systems with minimal modifications.

4.6 Comparative Performance with the Classical Computer vision Models

Compared with classical computer vision techniques such as background subtraction, motion detection, and handcrafted feature-based classifiers, the proposed system demonstrates several advantages. Traditional methods often struggle with complex backgrounds, dynamic lighting conditions, and variations in object appearance, which can lead to higher false positive rates and poor generalization.

In contrast, the deep learning–based approach used in this work provides robust feature extraction and class-specific detection capabilities. The ability to detect multiple object categories simultaneously and filter predictions based on confidence scores significantly improves detection reliability. Additionally, classical approaches often require extensive manual tuning and do not scale well across different environments, whereas the proposed system maintains stable performance without environment-specific adjustments.

4.7 Comparison of Existing Deep Learning Surveillance Systems

Compared with existing deep learning–based surveillance systems reported in the literature, the proposed system distinguishes itself through its emphasis on system-level integration and real-time usability. Many existing studies focus primarily on model accuracy and dataset-based benchmarking, while providing limited attention to real-time deployment, alert mechanisms, and user interaction.

The proposed system integrates detection, alert generation, visualization, and system control into a unified web-based platform. This end-to-end architecture enables instant feedback and remote monitoring capabilities. Furthermore, the use of a lightweight detection model ensures real-time performance without requiring specialized GPU hardware, providing a practical advantage over computationally intensive approaches.

4.8 System Scalability and Deployment Analysis

Scalability was analyzed by evaluating the system's ability to support extended operation and future expansion. The modular architecture of the proposed system allows straightforward extension to multiple camera streams, additional object classes, and customizable alert rules. The web-based interface supports remote access, enabling centralized monitoring from different locations.

The use of standard web technologies ensures compatibility across multiple devices and operating systems, improving deployment flexibility. Additionally, the system's moderate computational requirements allow horizontal scaling with minimal additional resources. These characteristics make the proposed solution suitable for applications ranging from small-scale surveillance systems to large distributed monitoring environments.

4.9 Practical Deployment and Scalability Analysis

Although the proposed system demonstrates strong performance in real-time operation, certain limitations were observed during experimentation. Detection accuracy may decrease under extreme lighting conditions or heavy object occlusion, which remains a common challenge for vision-based systems. Additionally, the use of a single RGB camera limits the system's ability to capture depth information.

The current implementation also focuses on a predefined set of obstacle classes, which may restrict its applicability in highly specialized environments. However, these limitations primarily relate to sensor and model constraints rather than the system architecture itself. Future improvements may include multimodal sensing, depth cameras, or updated object detection models to further enhance system capability.

4.10 Limitations and Actual Constraints

While the proposed system has shown good performance in real-time, some limitations were experienced during experimentation: Detection accuracy can be influenced under extreme lighting conditions or heavy occlusion, which is one of the major challenges for vision based systems. Additionally, the use of a single RGB camera is limiting during the depths and spatial tenacity.

Current implementation centres on a pre-defined set of obstacle classes, so this implementation may not be broadly applicable in highly specialised environments. However, these limitations are mostly related to model and sensors limitations and not related to system architecture and they can be solved by future improvements such as multimodal sensing or updates of the model.

Overall Findings Summary

The experimental results demonstrate that the proposed real-time object detection and alert system provides reliable detection performance, low-latency alerts, and stable operation under live monitoring conditions. Its efficient architecture, practical deployment capability, and integrated web-based interface make it well suited for real-time surveillance and safety monitoring applications.

5. Conclusion and Future Scope

5.1 Conclusion

This work introduced a complete real-time object detection and alert system with the perspective of continuous video surveillance and safety monitoring applications. By combining a lightweight YOLO-based deep learning network, OpenCV library and a Flask-based web framework the proposed system supports stable and real-time detection with computation efficiency. The system is effective in identifying the important obstacle such as Pedestrians and vehicles on site from live video streaming and immediate Audio-visual alerts from the vehicle, thus improving situational awareness and responsiveness among users.

The modular system architecture provides the ability to easily integrate video acquisition content, deep learning inference, alert decision logic and web-based visualization in a unified platform. Optimized frame processing and multithreaded processing ensures a smooth video streaming without the compromising detection accuracy. Experimental observations accounted for consistent performance under live operating condition, system alert latency and detection reliable performance using standard computing hardware.

Overall, the proposed solution is a solution to the main limitations of traditional surveillance systems, and it features an automated, responsive and simple to use monitoring structure. Its lightweight design and the ability to scale up makes it ideal to the deployment of real-world applications such as smart surveillance systems, assistive safety applications and intelligent monitoring platforms. Further improvements in this line can be made with multimodal sensing, adaptive model optimization and edge-based deployment capabilities to make the tool even robust and applicable.

5.2 Future Scope

Future improvements of the proposed system for real time object detection and hybrid alerts can be on the enhancement

of robustness, scalability and intelligence. The system can be adjusted by adding multimodal sensors such as depth cameras, infrared sensors or laser distance sensors (LiDAR) to increase the detection accuracy of the system in case of poor lighting and occlusion. Deployment on edge and embedded platform via model optimization techniques can minimize the latency and energy consumption. Additionally, adaptive learning and explainable AI mechanisms can be added to make the system more transparent and reliable. Expanding object classes and providing multi-camera shall increase the applicability for large-scale surveillance and smart safety environments even more.

References

[1] R. Gholami, H. D. Jahromi, and S. Sedaghat, "Design and implementation of a near real-time human detection robot using YOLO framework and IoT technologies," *IEEE Access*, 2025.
[2] M. P. Sudeshna, G. K. Karthika, K. Prathyusha, and K. Indhu, "Object detection and recognitions using webcams with voice using YOLO algorithm," *Environments*, vol. 25, no. 1, 2025.
[3] M. K. A. Maaroof and M. S. Bouhlel, "Real-time object detection using YOLOv8 model: A drone-based approach," *J. Wireless Mobile Netw., Ubiquitous Comput., Dependable Appl.*, vol. 16, no. 1, pp. 190–204, 2025.
[4] L. Marique, R. Delyn, and J. Limani, "Real-time object detection for smart city surveillance and traffic management systems," *ITEJ*, vol. 10, no. 1, pp. 153–159, 2025.
[5] D. Kaur, J. Singh, and N. K. Chahar, "Enhancing public safety through real-time video analytics using AI: A YOLOv8-based approach," 2025.
[6] M. Kotthapalli, D. Ravipati, and R. Bhatia, "YOLOv1 to YOLOv11: A comprehensive survey of real-time object detection innovations and challenges," *arXiv preprint arXiv:2508.02067*, 2025.
[7] R. Sapkota, R. H. Cheppally, A. Sharda, and M. Karkee, "YOLO26: Key architectural enhancements and performance benchmarking for real-time object detection," *arXiv preprint arXiv:2509.25164*, 2025.
[8] S. Ikram *et al.*, "Obstacle detection and warning system for visually impaired using IoT sensors," *IEEE Access*, 2025.
[9] C. L. Luo, H. Xu, M. Liu, and S. C. Chu, "An affordable intelligent navigation backpack for the visually impaired," *J. Internet Technol.*, vol. 26, no. 2, pp. 265–271, 2025.
[10] Z. Zheng *et al.*, "Real-time roadway obstacle detection for electric scooters using deep learning and multi-sensor fusion," *arXiv preprint arXiv:2504.03171*, 2025.
[11] K. H. Tang *et al.*, "Synchronization, optimization and adaptation of machine learning techniques for computer vision in cyber-physical systems," *IET Cyber-Phys. Syst.*, vol. 10, no. 1, 2025.
[12] A. E. R. Abd-El-Rehim, M. B. Badawi, and R. Elgohary, "Enhancing disaster response efforts with YOLOv8-based human detection in mobile robotics," *Int. Integrated Intelligent Syst.*, vol. 2, no. 1, 2025.
[13] Y. Vineeth, "Integrating computer vision and natural language processing for minimizing and detecting collisions," *Authorea Preprints*, 2025.
[14] Y. S. Dube, R. R. Shinde, and A. I. Chavhan, "Deep learning-based autonomous driving system with OpenCV integration," *IJSAT*, vol. 16, no. 2, 2025.
[15] M. S. K. Namana and B. U. Kumar, "Enhancing surveillance systems leveraging AIoT," *Eng., Technol. Appl. Sci. Res.*, vol. 15, no. 3, 2025.
[16] V. Godase, "Edge AI for smart surveillance," *Int. J. AI Mach. Learn. Innov. ECT*, vol. 1, no. 1, pp. 29–46, 2025.
[17] H. Khan *et al.*, "Violence detection from industrial surveillance videos using deep learning," *IEEE Access*, 2025.
[18] P. Vallurupalli and S. Belidhe, "Deploying lightweight neural networks on edge devices using knowledge transfer," 2025.
[19] M. Rahmati, "Towards explainable and lightweight AI for real-time cyber threat hunting," *arXiv preprint arXiv:2504.16118*, 2025.
[20] K. S. Kang *et al.*, "Generative infrared augmentation and lightweight detectors for nighttime infant safety monitoring," *SSRN*, 2025.
[21] I. S. Bas *et al.*, "CNN transfer learning method for aircraft image classification," 2025.
[22] Y. Zhu, X. Niu, and J. Tian, "Machine vision-based intelligent segmentation for dam underwater cracks," *Comput.-Aided Civ. Infrastruct. Eng.*, vol. 40, no. 10, 2025.
[23] Y. Ma *et al.*, "Machine vision-based intelligent tracking system for maintenance personnel," *J. Adv. Transp.*, 2025.
[24] M. Alshehri *et al.*, "UAV-based multi-person detection via deep neural networks," *Front. Neurorobot.*, vol. 19, 2025.
[25] S. R. Alotaibi *et al.*, "Integrating explainable AI for crowd density estimation," *IEEE Access*, 2025.
[26] W. Zhang *et al.*, "Multimodal deep learning-based intelligent food safety detection," *Int. J. Manage. Sci. Res.*, vol. 8, no. 3, 2025.
[27] S. Ataei *et al.*, "Data-driven damage detection in concrete structures," *arXiv preprint arXiv:2501.11836*, 2025.
[28] M. A. Putra, A. Harjoko, and Wahyono, "Vision-based traffic density estimation: A review," *IET Intell. Transp. Syst.*, vol. 19, no. 1, 2025.
[29] S. P. S. Nithin *et al.*, "Revolutionizing industrial safety with real-time video analytics," *IJSAT*, vol. 16, no. 1, 2025.
[30] S. Taware *et al.*, "Video sentiment analysis using deep learning," 2025.
[31] S. M. Sakthi *et al.*, "Real-time passenger counting using YOLO," *J. Mach. Learn. Signal Process.*, 2025.
[32] D. Murthy *et al.*, "AI-driven disaster damage surveillance," *Int. J. Disaster Risk Reduct.*, vol. 116, 2025.
[33] S. S. Maharajpet *et al.*, "AI-driven hostel monitoring system," *Eur. J. Appl. Sci. Eng. Technol.*, vol. 3, no. 3, 2025.
[34] A. Hamza *et al.*, "Autonomous AI surveillance using multimodal deep learning," *arXiv preprint arXiv:2507.01590*, 2025.

[35] M. Alymani *et al.*, "Enabling smart parking for smart cities using IoT and machine learning," *PeerJ Comput. Sci.*, vol. 11, e2544, 2025.
[36] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified Real-Time Object Detection," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[37] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv preprint arXiv:2004.10934, 2020.