ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 1, pp: (122-127), Month: January - March 2026, Available at: www.researchpublish.com

# Visualizing and Processing Streaming Media with Object Identification Using OpenCV

[1]Sujan C R, [2]Veeresh I Hiremath, [3]Vishal K, [4]Deepika D Pai

[1,2,3] Students,[4]Assistant Professor

[1,2,3,4]Vemana Institute of Technology, Bengaluru, India

**DOI: https://doi.org/10.5281/zenodo.18836078**

**Published Date: 02-March-2026**

***Abstract:*** **This paper presents a real-time object detection system for visualizing and processing streaming media using OpenCV and the YOLOv5 deep learning model. The proposed system captures live video streams and processes each frame to accurately detect and classify multiple objects in real time. OpenCV is used for video acquisition, preprocessing, and visualization, while YOLOv5 performs fast and reliable object detection by generating bounding boxes and class labels. The system effectively detects objects such as persons, mobile phones, and clocks with high accuracy and smooth frame rates. Experimental results demonstrate that the proposed approach achieves a good balance between detection accuracy and real-time performance on standard laptop hardware. The developed system is scalable and can be extended to advanced applications such as surveillance, traffic monitoring, and intelligent video analytics.**

***Keywords:*** **OpenCV, YOLOv5, Object Detection, Computer Vision, Deep Learning, Real-Time Video Processing, Image Processing.**

## I. INTRODUCTION

In recent years, the rapid growth of digital cameras, surveillance systems, and intelligent applications has significantly increased the demand for efficient real-time video analysis. Manual monitoring of continuous video streams is inefficient, error-prone, and impractical for large-scale environments. As a result, automated object detection using computer vision and deep learning techniques has gained considerable attention. Object detection aims to identify and localize multiple objects within images or video frames by assigning class labels and bounding boxes, enabling machines to understand visual scenes effectively.

Traditional object detection approaches such as Haar Cascade classifiers and HOG-based methods are computationally efficient but perform poorly in complex lighting conditions, cluttered backgrounds, and crowded scenes. SSD-based detectors provide reasonable real-time performance but often suffer from reduced accuracy when compared to modern YOLO-based architectures. Earlier versions of YOLO, such as YOLOv3 and YOLOv4, improved detection speed but still faced limitations in small-object detection and bounding box stability. In contrast, YOLOv5 offers a superior balance between speed and accuracy due to its optimized architecture, advanced data augmentation techniques, and improved loss functions. Studies have shown that YOLOv5 achieves higher mean Average Precision (mAP), better small-object detection, and more stable bounding box predictions than previous models. When combined with OpenCV, YOLOv5 demonstrates enhanced robustness and real-time performance in practical applications such as surveillance, traffic monitoring, and pedestrian detection.

Despite these advancements, several research challenges remain. Existing systems still struggle with detecting small objects and maintaining accuracy in crowded environments. Many previous works fail to achieve both high accuracy and real-time performance on low-power or resource-constrained devices. Additionally, limited research has focused on integrating advanced tracking algorithms with YOLOv5 to preserve object identity across consecutive frames. Deployment on edge

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 1, pp: (122-127), Month: January - March 2026, Available at: www.researchpublish.com

devices such as Raspberry Pi and NVIDIA Jetson platforms also remains underexplored. Furthermore, handling low-light conditions, occlusions, and noisy backgrounds continues to be a challenge. These gaps highlight the need for a fast, scalable, and accurate object detection framework that effectively combines OpenCV-based preprocessing with YOLOv5 detection to improve real-time performance in real-world environments.

## II. METHODOLOGY

The methodology of the proposed system focuses on real-time object detection from streaming video using OpenCV and the YOLOv5 deep learning model. The system begins by capturing live video frames from a webcam or video source using OpenCV. Each captured frame is resized and normalized to match the input requirements of the YOLOv5 model.The preprocessed frames are then passed to the YOLOv5 model, which performs object detection in a single forward pass. The model predicts bounding box coordinates, confidence scores, and class labels for multiple objects present in each frame. To improve detection accuracy, confidence thresholding and Non-Maximum Suppression (NMS) are applied to remove low-confidence and overlapping detections.

Finally, OpenCV is used to draw bounding boxes, class labels, and confidence scores on the detected objects and display the output in real time. The entire process is repeated for every frame, enabling continuous and efficient real-time object detection suitable for practical applications.

The object detection process is mathematically modeled using bounding box prediction, confidence score calculation, and Non-Maximum Suppression (NMS). The YOLOv5 model predicts bounding boxes in terms of center coordinates, width, and height, which are converted into corner coordinates for visualization. The confidence score is computed based on objectness probability and class probability. Intersection over Union (IoU) is used to suppress overlapping bounding boxes, ensuring accurate localization of detected objects.

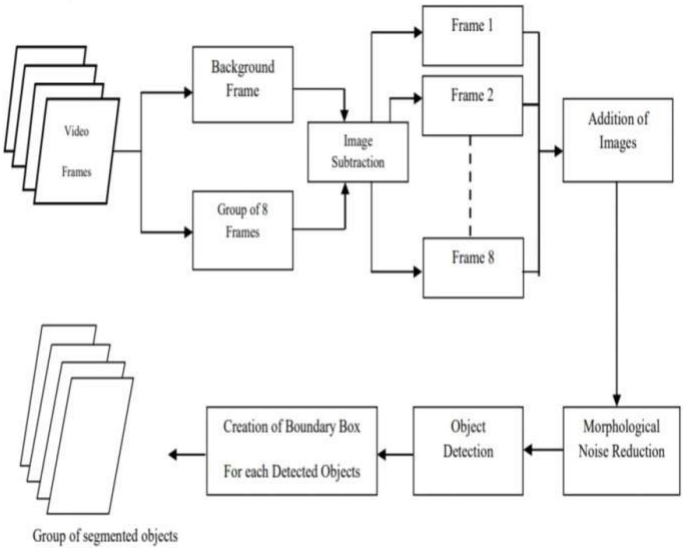


**Fig. 1. Block diagram of object detection using OpenCV and YOLOv5**

The system for visualizing and processing streaming media with object identification using OpenCV can be implemented through a series of structured steps:

**Data Acquisition**: The first step involves collecting data either through real-time video feeds using cameras or webcams, or by leveraging publicly available datasets containing images or videos relevant to the target objects. This ensures a continuous stream of data for processing and evaluation, which is crucial for real-time applications like surveillance or traffic monitoring.

**Preprocessing**: Once the video frames are acquired, preprocessing is performed to enhance the quality and suitability of the input data for detection algorithms. This step typically involves resizing the images to a standard resolution to reduce computational load, normalizing pixel values to a consistent scale for better model performance, and converting color channels (for example, from RGB to grayscale) as required by specific detection methods. Optional preprocessing steps include de-noising using filters or applying blurring techniques to reduce image noise and enhance feature extraction.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 1, pp: (122-127), Month: January - March 2026, Available at: www.researchpublish.com

**Object Detection**: Object detection is performed using a combination of traditional image processing methods and modern deep learning models. Basic pixel-level operations and filters in OpenCV can identify regions of interest, while advanced detection models such as YOLO (You Only Look Once) or SSD (Single Shot MultiBox Detector) can detect and classify multiple objects in a single frame with high accuracy. OpenCV serves as the integration platform to manage these detection pipelines efficiently.

**Object Tracking**: To maintain the identity of detected objects across consecutive frames, tracking algorithms are employed. Techniques such as Euclidean distance-based tracking, Kalman filters, or more advanced optical flow methods can be used to ensure that each object is continuously monitored throughout the video stream. This step is crucial for applications like pedestrian tracking, vehicle monitoring, or security surveillance.

**Output Visualization**: Detected objects are highlighted in the live video feed using bounding boxes, along with class labels to indicate the type of object detected. This provides an intuitive visualization for monitoring and analysis. Additionally, optional performance metrics, such as frames per second (FPS), can be displayed to evaluate the efficiency and responsiveness of the system in real time.

**System Optimization and Feedback**: Depending on the application requirements, additional steps can include fine-tuning detection thresholds, optimizing frame processing rates, or integrating feedback loops to improve detection accuracy and reduce false positives.

By following this methodology, the system can efficiently process streaming media, detect and track objects in real-time, and provide clear visual feedback for monitoring, analysis, and decision-making in various domains such as intelligent vehicles, robotics, and security systems.

*1) Mathematical Equations Used in Object Detection*

**1. Image Normalization-** Before feeding the image to the YOLOv5 model, pixel values are normalized to improve learning efficiency:

where, $I$= original pixel value (0–255).

$$I_{norm} = \frac{I}{255}$$

**2.** *Bounding Box Conversion*

YOLOv5 predicts bounding boxes in center format ($c_x$, $c_y$, , $h$). These are converted to corner coordinates as:

$$x_{min} = c_x - \frac{w}{2}$$
$$y_{min} = c_y - \frac{h}{2}$$
$$x_{max} = c_x + \frac{w}{2}$$
$$y_{max} = c_y + \frac{h}{2}$$

These coordinates are then scaled to the original image dimensions.

**3. Confidence Score Calculation-** The final confidence score for each detected object is computed as:

$$Confidence = P_{obj} \times P(class)$$

Where, $P_{obj}$= objectness probability

($class$)= class probability.

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 1, pp: (122-127), Month: January - March 2026, Available at: www.researchpublish.com

4. **Intersection over Union (IoU)-** IoU measures the overlap between predicted and ground truth bounding boxes:

*B. IoU= Area of overlap/area of union*

A higher IoU indicates better localization accuracy.

5. **Non-Maximum Suppression (NMS) Condition-** To remove duplicate detections:

$$I(b_i, b_j) > T_{nms} \Rightarrow \text{Suppress } b_j$$

Where, $T_{nms}$= NMS threshold.

6. **Frame Rate (FPS) Calculation-** Real-time performance is measured using Frames Per Second:

*C. FPS= 1/Time per frame*

Higher FPS indicates smoother real-time detection.

7. **Class Selection-** The detected class is selected using:

$$Class\_ID = \arg\max((class_1), (class_2), \dots, P(class_n))$$

## III. RESULTS AND DISCUSSION

The proposed object detection system was tested using live video streams captured through a webcam. The YOLOv5 model successfully detected multiple objects in real time, including persons, mobile phones, and clocks. Each detected object was highlighted with a bounding box along with its corresponding class label and confidence score, demonstrating accurate localization and classification. The mathematical modeling techniques such as confidence thresholding, IoU-based Non-Maximum Suppression, and bounding box conversion significantly improved detection accuracy and reduced false positives in real-time detection.

The system achieved smooth real-time performance with stable frame rates on standard laptop hardware. The application of confidence thresholding and Non-Maximum Suppression effectively reduced false detections and overlapping bounding boxes. Compared to traditional OpenCV-based detection methods, the YOLOv5-based approach provided higher accuracy, better robustness under varying conditions, and faster detection speed.The experimental results confirm that integrating OpenCV with YOLOv5 offers a reliable and efficient solution for real-time object detection in streaming media, making it suitable for applications such as surveillance, monitoring, and intelligent video analysis.

**Design Specification:** The proposed object detection system employs OpenCV and the YOLOv5 model to perform real-time analysis of streaming video obtained from a webcam or IP camera. Input frames are preprocessed using resizing and pixel normalization before being passed to the YOLOv5s detector, which is selected to balance accuracy and computational efficiency on standard laptop hardware. Object detection is refined using confidence thresholding and IoU-based Non-Maximum Suppression to eliminate redundant detections and improve localization accuracy. The system produces annotated video output with bounding boxes, class labels, confidence scores, and real-time FPS visualization, enabling effective monitoring and analysis.Performance evaluation is carried out using metrics such as frame rate, inference latency, Precision, Recall, and mean Average Precision (mAP), demonstrating reliable real-time performance. The model is initialized with pretrained COCO dataset weights and supports extension to custom datasets annotated in YOLO format, allowing adaptability to domain-specific applications. Functional, performance, and robustness testing indicate stable operation under moderate illumination changes, partial occlusion, and background complexity. The modular design of the system allows seamless integration with tracking algorithms and future enhancements, confirming its suitability for real-world surveillance, traffic monitoring, and intelligent video analytics applications.

**Advantages:** The proposed object detection system using OpenCV and YOLOv5 offers several advantages. YOLOv5 is optimized for high-speed inference, enabling real-time object detection with smooth frame rates even on standard laptop hardware. The model achieves high accuracy and robustness, effectively detecting multiple objects under varying lighting conditions and complex backgrounds. Since YOLOv5 performs detection in a single forward pass, computational overhead and processing delays are significantly reduced. The availability of pretrained models and support for custom training make the system adaptable to different application domains. Additionally, the integration of YOLOv5 with OpenCV simplifies video capture, preprocessing, and visualization. The system also supports simultaneous detection of multiple object classes

ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 14, Issue 1, pp: (122-127), Month: January - March 2026, Available at: www.researchpublish.com

and is cross-platform compatible, making it suitable for applications such as surveillance, traffic monitoring, and intelligent video analytics.

**Applications:** The developed object detection system using OpenCV and YOLOv5 can be applied to a wide range of real-world scenarios. It is highly suitable for smart surveillance systems, enabling real-time monitoring of public spaces, campuses, and commercial environments by detecting human presence and suspicious activities. In traffic monitoring applications, the system can be used for vehicle detection, traffic flow analysis, congestion monitoring, and violation detection. The approach also supports smart city applications by facilitating intelligent video analytics for public safety and crowd management. Furthermore, the system can be extended to autonomous vehicles and drones for detecting pedestrians, obstacles, and road signs, as well as to wildlife monitoring systems for tracking animal movement and preventing illegal activities.

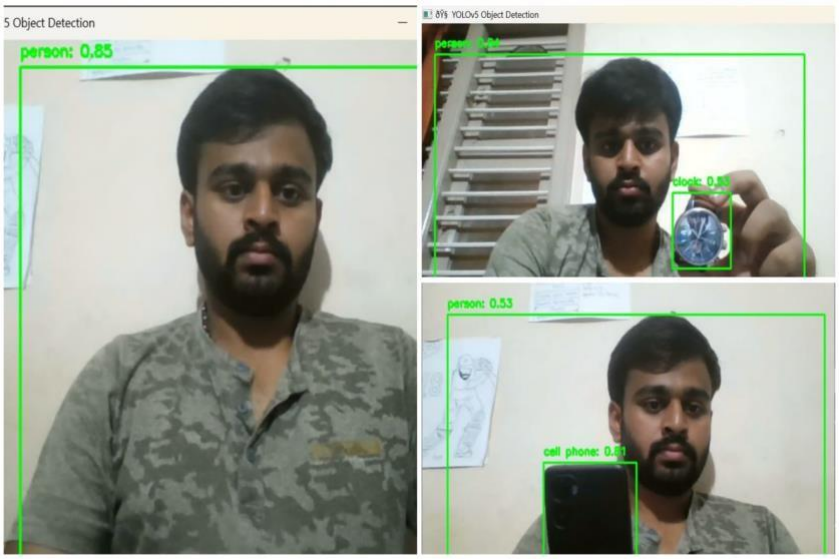


**Fig. 2. Result**

**Result Description:** The real-time object detection system successfully identified multiple objects such as person, cell phone, and clock etc… from the live feed. As seen in the output frames, the YOLOv5 model accurately detected each object and marked them with green bounding boxes along with corresponding class labels and confidence scores. The system also maintained smooth processing by displaying the FPS and updating detections continuously for every frame. This demonstrates the effective performance of the OpenCV + YOLOv5 pipeline in detecting and classifying objects with high accuracy in real-time conditions.

## IV. CONCLUSION

The object detection system developed using OpenCV and the YOLOv5 model successfully fulfills the intended design objectives by providing accurate, fast, and real-time detection across multiple object categories. The underlying mathematical components—including image normalization, bounding box conversion, confidence scoring, and IoU-based Non-Maximum Suppression—ensure stable and reliable detection throughout the processing pipeline. Experimental results demonstrate consistent detection of objects such as persons, mobile phones, and clocks with precise bounding boxes, confidence scores, and smooth FPS performance. Compared to traditional OpenCV-based methods and earlier YOLO versions, YOLOv5 exhibits superior accuracy, robustness, and real-time processing capability.

Furthermore, the integration of OpenCV for video acquisition, preprocessing, and visualization simplifies system implementation while maintaining high efficiency. The proposed approach effectively balances computational complexity and detection performance, making it suitable for deployment on standard laptop hardware without the need for high-end resources. The modular architecture of the system allows easy customization, scalability, and integration with additional components such as object tracking, alert mechanisms, and cloud-based analytics. Overall, the system proves to be efficient, scalable, and adaptable, providing a strong foundation for real-world applications including surveillance, monitoring, and intelligent video analytics, as well as for future research involving advanced deep learning models, edge deployment, and large-scale intelligent vision systems.

**Future Scope:** The proposed system can be further extended by incorporating action and behavior recognition to identify human activities and suspicious events for enhanced security applications. Vehicle analytics such as speed estimation, license plate recognition, and traffic flow analysis can be integrated to support intelligent transportation systems. Future work may also focus on optimizing the model for deployment on edge devices like Raspberry Pi and Jetson Nano, enabling scalable and cost-effective real-time monitoring. Additionally, support for multi-camera environments can enhance smart city and public safety applications. The system can be upgraded with advanced vision techniques such as image segmentation and pose estimation, along with real-time alert mechanisms and privacy-preserving features, making it suitable for large-scale AI-driven surveillance and analytics platforms.

## REFERENCES

[1] Sharma, A., et al., "Object Detection using OpenCV and Python," International Journal of Computer Applications, 2021.

[2] Dardagan, S., et al., "Multiple Object Trackers in OpenCV: A Benchmark," IEEE Access, 2021.

[3] Terven, J., and Córdova-Esparza, D., "A Comprehensive Review of YOLO Architectures: From YOLOv1 to YOLOv8 and YOLO-NAS," Journal of Computer Vision, 2023.

[4] S. Siva, et al., "Real-Time Object Detection using OpenCV," International Journal of Engineering Research, 2023.

[5] Swathi Srikanth Achanur, et al., "Object Detection Based on Python and Image Processing," International Research Journal of Engineering and Technology, 2025.