Design And Implementation of A Real-Time Multi-Object Detection And Tracking System Using YOLO26

Reagan Joseph S[1], Santhosh B[2], Sabari KrishnanK[3], Ajina H[4]
[1, 2, 3, 4]Dept of Artificial intelligence and Data Science,
[1, 2, 3, 4]Rathinam Technical Campus

Abstract- *This paper presents the development of a real-time object detection and multi-object tracking system using the Ultralytics YOLO26 model. The system captures live video through a webcam and processes each frame using a deep learning-based computer vision model to detect multiple objects simultaneously. The detected objects are highlighted with bounding boxes and assigned unique tracking IDs to maintain their identity across consecutive frames. The system is implemented using Python, OpenCV, and the PyTorch framework. The proposed model is executed on CPU hardware without GPU acceleration and achieves an average processing speed of approximately 18–25 frames per second. Experimental results show that the system can detect common objects such as persons, vehicles, and everyday items effectively in real time. The proposed approach provides a cost-effective and efficient solution for surveillance systems, smart monitoring, and intelligent vision-based applications.*

Keywords: Computer Vision, Deep Learning, Multi-Object Tracking, Object Detection, YOLO

I. INTRODUCTION

Computer vision is a rapidly evolving field of artificial intelligence that enables machines to interpret and analyze visual information from images and videos. One of the most important applications of computer vision is object detection, which involves identifying and locating objects in real-time environments. Object detection systems are widely used in applications such as security surveillance, traffic monitoring, autonomous vehicles, and smart retail systems.

Traditional object detection techniques relied on manual feature extraction methods, which required significant computational effort and were less efficient in complex environments. With the emergence of deep learning and convolutional neural networks (CNNs), object detection systems have become more accurate and efficient.

The YOLO (You Only Look Once) algorithm introduced a single-stage object detection approach that performs both object localization and classification in a single pass of the neural network. This significantly improves detection speed, making it suitable for real-time applications.

IDENTIFY, RESEARCH AND COLLECT IDEA

The first step in developing the proposed system involvedidentifying the research problem and understanding existing object detection techniques. Several research papers and technical resources related to computer vision, deep learning, and object detection were studied to understand the working principles of modern detection algorithms.

Existing models such as R-CNN, Faster R-CNN, and SSD were analyzed to understand their advantages and limitations. Among these approaches, the YOLO algorithm demonstrated superior performance in terms of detection speed and real-time capability. Therefore, the Ultralytics YOLO26 model was selected for this project due to its efficiency and improved accuracy.

The research process also included studying various computer vision libraries such as OpenCV and deep learning frameworks like PyTorch. These tools were used to implement the detection model and process real-time video frames effectively.

Read Already Published Work in the Same Field

To understand the existing research in the field of object detection and computer vision, several previously published research papers and journal articles were studied. These papers provided insights into different object detection algorithms such as R-CNN, Faster R-CNN, SSD, and YOLO. By reviewing these studies, the advantages and limitations of each method were analyzed. Most recent research emphasizes the use of deep learning–based models for improving detection accuracy and speed. The YOLO algorithm was identified as one of the most efficient real-time object detection models due to its single-stage detection architecture and high processing speed. This analysis helped in selecting the Ultralytics YOLO model as the foundation for the proposed system.

Googling on the Topic of Research Work

In addition to reviewing research papers, extensive online research was conducted to gain practical knowledge about implementing real-time object detection systems. Various technical resources, tutorials, documentation, and open-source repositories were explored through online search engines. Platforms such as GitHub, technical blogs, and official documentation for OpenCV and PyTorch provided guidance on installing libraries, configuring the YOLO model, and processing video frames for detection tasks. This online research helped in understanding the implementation process and provided practical solutions for developing the real-time object detection and multi-object tracking system.

II. WRITE DOWN YOUR STUDIES AND FINDINGS

After collecting sufficient knowledge and resources, the system was implemented using Python programming language. The webcam was used as the input device to capture live video frames. Each frame was processed by the YOLO26 object detection model to identify objects and draw bounding boxes around them.

The model assigns class labels and confidence scores to detected objects and maintains tracking IDs to monitor multiple objects across frames. The results show that the proposed system is capable of detecting multiple objects simultaneously with stable tracking performance.

The implementation demonstrated that real-time object detection can be achieved effectively even on CPU-based systems without requiring high-end GPU hardware. The system provides reliable detection results and can be further enhanced for advanced applications such as automated surveillance systems, smart traffic monitoring, and intelligent video analytics.

III. CONCLUSION

In this research work, a real-time object detection and multi-object tracking system was successfully developed using the Ultralytics YOLO26 model. The system processes live video frames captured from a webcam and detects multiple objects simultaneously by generating bounding boxes, class labels, and confidence scores. Each detected object is assigned a unique tracking ID to maintain object identity across consecutive frames.

The implementation was carried out using Python, OpenCV, and the PyTorch framework on CPU hardware without GPU acceleration. Experimental results demonstrated that the system can achieve an average processing speed of approximately 18–25 frames per second while maintaining stable detection performance.

The proposed system provides a practical and efficient solution for real-time computer vision applications such as surveillance systems, traffic monitoring, and smart security environments. In the future, the system can be further improved by integrating GPU acceleration, training the model on custom datasets, and deploying the system in large-scale intelligent monitoring applications.

APPENDIX

The appendix section may include additional implementation details related to the developed system. This includes configuration parameters used in the YOLO26 model, software libraries required for execution, and system requirements for running the application.

The system was developed using Python programming language along with libraries such as OpenCV for video processing and PyTorch for deep learning model execution. The application captures live video input from a webcam and processes each frame through the YOLO26 detection model to identify objects in real time.

Additional experiment screenshots and output samples are also included to demonstrate the performance of the object detection and tracking system.

IV. ACKNOWLEDGMENT

The authors would like to express their sincere gratitude to the Department of Artificial Intelligence and Data Science at Rathinam Technical Campus for providing the necessary guidance and resources to carry out this research work. The authors also thank the project guide and faculty members for their continuous support, valuable suggestions, and encouragement throughout the development of this project.

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,”

Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[2] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[3] A. Bochkovskiy, C. Wang, and H. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” 2020.

[4] C.-Y. Wang, A. Bochkovskiy, and H.-Y. Liao, “YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors,” 2022.

[5] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE TPAMI, 2017.

[6] J. Long, E. Shelhamer, and T. Darrell, “Fully Convolutional Networks for Semantic Segmentation,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] A. Geiger, P. Lenz, and R. Urtasun, “Are We Ready for Autonomous Driving? The KITTI Vision Benchmark Suite,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[9] T. Lin et al., “Microsoft COCO: Common Objects in Context,” *European Conference on Computer Vision (ECCV)*, 2014.

[10] J. Deng et al., “ImageNet: A Large-Scale Hierarchical Image Database,” *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.