iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VII **Month of publication:** July 2026

DOI: https://doi.org/10.22214/ijraset.2026.84315

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

Object Detection and Tracking, Searching Using Computer Vision and YOLO Technique

Banka Vinay[1], Mrs. Pravitha R. Prasad[2]
[1]*Student, Master of Computer Applications (MCA), AQJ Centre for P.G. Studies (Affiliated to Andhra University, Visakhapatnam), Visakhapatnam, Andhra Pradesh, India*
[2]*Assistant Professor, Department of MCA, AQJ Centre for P.G. Studies (Affiliated to Andhra University, Visakhapatnam), Visakhapatnam, Andhra Pradesh, India*

Abstract: ***Real-time visual understanding has become a cornerstone of modern intelligent systems, spanning surveillance, autonomous navigation, retail analytics, and smart-city infrastructure. The system architecture is described in detail, including data-flow diagrams, use-case and sequence diagrams, an activity diagram, and a relational database design for persisting detection and tracking metadata. An experimental evaluation performed on standard benchmark-style data demonstrates that the proposed pipeline attains a mean Average Precision (mAP@0.5) of approximately 0.91, a tracking identity-switch rate reduced by 38% relative to a naive frame-by-frame detector, and a sustained throughput of 61 frames per second on a mid-range GPU, outperforming several baseline architectures compared in this study. The results confirm that combining YOLOv8 with Deep SORT and a dedicated search layer yields a practical, extensible platform for real-time object detection, tracking, and retrieval applications.***

Keywords: Object Detection, Object Tracking, Object Searching, Computer Vision, YOLOv8, Deep SORT, Convolutional Neural Network, Real-Time Video Analytics, Deep Learning, Surveillance Systems.

I. INTRODUCTION

The rapid growth of digital imaging, affordable high-resolution cameras, and inexpensive computational hardware has fuelled an explosion of interest in automated visual analysis. Applications ranging from intelligent video surveillance and traffic monitoring to autonomous vehicles, retail loss-prevention, warehouse automation, and human-computer interaction all rely on a common underlying capability: the ability to reliably locate, classify, and follow objects of interest within images and video streams, and subsequently to retrieve or search for specific instances on demand. This capability is collectively addressed by the fields of object detection, multi-object tracking, and content-based visual search, all of which fall under the broader discipline of computer vision.

Object detection is the task of simultaneously localizing (via bounding boxes) and classifying every instance of a predefined set of object categories present in an image. Historically, detection pipelines were built using hand-crafted feature descriptors such as Haar cascades, Histogram of Oriented Gradients (HOG), and Scale-Invariant Feature Transform (SIFT) combined with classical classifiers such as Support Vector Machines. While effective for constrained problems, these approaches generalized poorly to cluttered scenes, varying illumination, occlusion, and scale variation. The advent of deep Convolutional Neural Networks (CNNs) fundamentally transformed the field, giving rise to region-proposal-based detectors such as R-CNN, Fast R-CNN, and Faster R-CNN, which achieved substantially higher accuracy but at the cost of multi-stage pipelines that are computationally expensive and difficult to run in real time.

This paper proposes and documents an integrated system that combines YOLOv8-based object detection, Deep SORT-based multi-object tracking, and a computer-vision-driven object searching module into a single, cohesive application supporting three input modalities — static images, pre-recorded video files, and live camera feeds. The remainder of this paper is organized as follows: Section II reviews related literature; Sections III and IV describe existing systems and their limitations;

Section V presents the proposed system and Section VI its objectives; Section VII details the methodology; Section VIII presents the system design; Section IX discusses implementation modules; Section X reports experimental results; and Sections XI-XIV discuss advantages, applications, future scope, and conclusions respectively.

II. LITERATURE SURVEY

A substantial body of research underpins the object detection, tracking, and retrieval pipeline proposed in this paper, summarized below by theme.

Region-proposal-based detectors: Girshick et al. introduced R-CNN, which used selective search to generate region proposals classified by a CNN; while accurate, it was too slow for real-time use. Fast R-CNN shared convolutional computation across proposals, and Faster R-CNN replaced selective search with a learned Region Proposal Network (RPN), improving speed while retaining high accuracy. These two-stage detectors remain influential benchmarks but are generally unsuitable for real-time video without significant hardware acceleration. Computer vision pre-processing: Classical techniques including frame differencing, background subtraction (Gaussian Mixture Models, ViBe), optical flow, and colour-space transformations continue to support modern pipelines for motion cueing and post-processing. OpenCV remains the dominant library for implementing these operations efficiently.

III. EXISTING SYSTEM

Existing video analytics deployments typically fall into one of three categories. The first category comprises manual or semi-automated surveillance review, in which security personnel visually monitor live camera feeds or scrub through recorded footage to identify events or objects of interest. This approach is labour-intensive, prone to human fatigue and error, and does not scale beyond a small number of simultaneous camera feeds.

The third category comprises deep-learning-based systems built around two-stage detectors such as Faster R-CNN or Mask R-CNN. These systems achieve high detection accuracy but incur substantial computational cost per frame, making them difficult to deploy for real-time, multi-stream video analysis without expensive GPU infrastructure. Furthermore, most existing systems address detection in isolation and either omit multi-object tracking entirely or implement tracking using simple centroid-distance heuristics that are unreliable under occlusion. Object searching, where supported at all, is usually limited to keyword-based metadata search rather than direct visual content matching, and rarely integrates tracking identities with search results.

IV. PROBLEMS IN EXISTING SYSTEM

Analysis of the existing systems described above reveals several recurring limitations that motivate the proposed work.

1) High computational latency: Two-stage detectors and heavyweight backbones impose significant per-frame inference cost, precluding real-time operation on standard hardware or requiring costly GPU clusters for multi-camera deployments.

2) Poor robustness under occlusion and clutter: Classical background-subtraction and heuristic tracking methods frequently lose object identity when objects overlap, cross paths, or are briefly occluded, resulting in fragmented or duplicated tracks.

These shortcomings collectively motivate the design of a system that (i) uses a fast, accurate single-stage detector, (ii) incorporates appearance-aware multi-object tracking to preserve identity under occlusion, (iii) supports images, video files, and live camera streams within one interface, (iv) provides a dedicated object-searching module operating directly on detection and tracking metadata, and (v) persists results in a structured database to enable reporting and analytics.

V. PROPOSED SYSTEM

The proposed system is an integrated computer-vision platform that unifies object detection, multi-object tracking, and object searching into a single workflow. At its core, the system employs YOLOv8 for fast and accurate object detection across static images, video files, and live camera streams. Detected bounding boxes are passed to a Deep SORT tracking module, which assigns and maintains persistent track identifiers for each object across frames using a combination of Kalman-filter-based motion prediction and a learned appearance embedding for re-identification.

All detection and tracking outputs — including class labels, confidence scores, bounding-box coordinates, track identifiers, and timestamps — are persisted to a relational database. A dedicated object-searching module allows users to query this database (and, where applicable, the live processing pipeline) using object class, approximate attributes, or a specific track identifier, returning matching frames, timestamps, and trajectories. A reporting module aggregates detection and search statistics into summary dashboards and exportable reports.

VI. OBJECTIVES

1) Design and implement a real-time object detection module using the YOLOv8 architecture capable of processing static images, recorded video files, and live camera streams with high accuracy and low latency.

2) Integrate a robust multi-object tracking mechanism using the Deep SORT algorithm to assign and maintain consistent object identities across frames, minimizing identity switches during occlusion or crossing trajectories.

3) Develop a computer-vision pre-processing pipeline for frame extraction, normalization, and enhancement to improve detection reliability under varying lighting and camera conditions.

ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

VII. METHODOLOGY

The methodology adopted in this work follows a modular pipeline comprising four principal stages: (1) a computer-vision pre-processing stage responsible for acquiring and normalizing frames from images, videos, or live camera feeds; (2) an object-detection stage built on the YOLOv8 architecture; (3) a multi-object tracking stage using the Deep SORT algorithm; and (4) an object-searching stage that indexes and retrieves detected and tracked objects on demand. Each stage is described in detail in the following subsections.

A. YOLOv8 Architecture

YOLOv8 is a single-stage, anchor-free object detector that predicts bounding boxes, objectness, and class probabilities directly from an input image in a single forward pass through the network. The architecture is organized into three principal components: a backbone, a neck, and a detection head.

The backbone is a CSPDarknet-style convolutional network that extracts hierarchical feature representations at multiple spatial resolutions. It employs Cross-Stage Partial (CSP) connections and the C2f (Cross-Stage Partial with two convolutions, fused) module, which splits feature maps into two branches — one that passes through a series of bottleneck blocks and one that bypasses them — before concatenating and fusing the outputs. This design improves gradient flow during training and reduces redundant computation relative to earlier CSPDarknet variants used in YOLOv4 and YOLOv5.

B. Deep SORT Tracking

Multi-object tracking in the proposed system is performed using the Deep SORT algorithm, which extends the original SORT framework by incorporating a learned appearance descriptor alongside motion-based association. The tracking pipeline operates as follows.

For each detected object in a frame, a bounding box and class label are received from the YOLOv8 detection stage. A Kalman filter models the object's motion state (position, velocity, and bounding-box scale/aspect ratio) and predicts its expected location in the subsequent frame. When new detections arrive, the Mahalanobis distance between predicted and observed states provides a motion-based association cost.

This combination of motion prediction and appearance re-identification allows the tracker to maintain consistent object identities across occlusion events, temporary detection failures, and crossing trajectories, substantially reducing the identity-switch rate relative to motion-only tracking approaches such as plain SORT or centroid-distance heuristics.

C. Computer Vision Pipeline

The computer-vision layer underpinning the proposed system is responsible for all pre-processing and post-processing operations surrounding the core detection and tracking models. On the input side, frames are acquired from static images, decoded video files, or a live camera stream using standard video I/O libraries (OpenCV). Each frame is resized and letterboxed to the fixed input resolution expected by the YOLOv8 network (typically 640 x 640 pixels), normalized to the [0, 1] range, and converted to the tensor format expected by the inference engine.

Optional pre-processing enhancements include contrast-limited adaptive histogram equalization (CLAHE) for low-light footage, denoising filters for compressed or noisy video streams, and colour-space conversions where attribute-based search (e.g., colour matching) is required. Frame-rate control logic ensures that live-camera processing keeps pace with the incoming stream by adaptively skipping frames if inference falls behind real time, while video-file processing can operate at the maximum throughput supported by the hardware.

D. Object Searching Module

The object-searching module enables users to retrieve specific objects or events from processed image, video, or live-stream data without manually reviewing footage. Three query modes are supported. Class-based search allows a user to request all instances of a given object category (e.g., 'car', 'person') detected within a specified time range or source. Attribute-based search extends class-based search with simple visual attributes such as approximate colour, derived from the dominant colour histogram of the cropped object region, allowing queries such as 'person wearing red'. Identity-based search allows a user to retrieve the complete trajectory and appearance history of a specific tracked object given its track identifier, which is particularly useful for forensic review of a single subject across an extended video sequence.

VIII. SYSTEM DESIGN

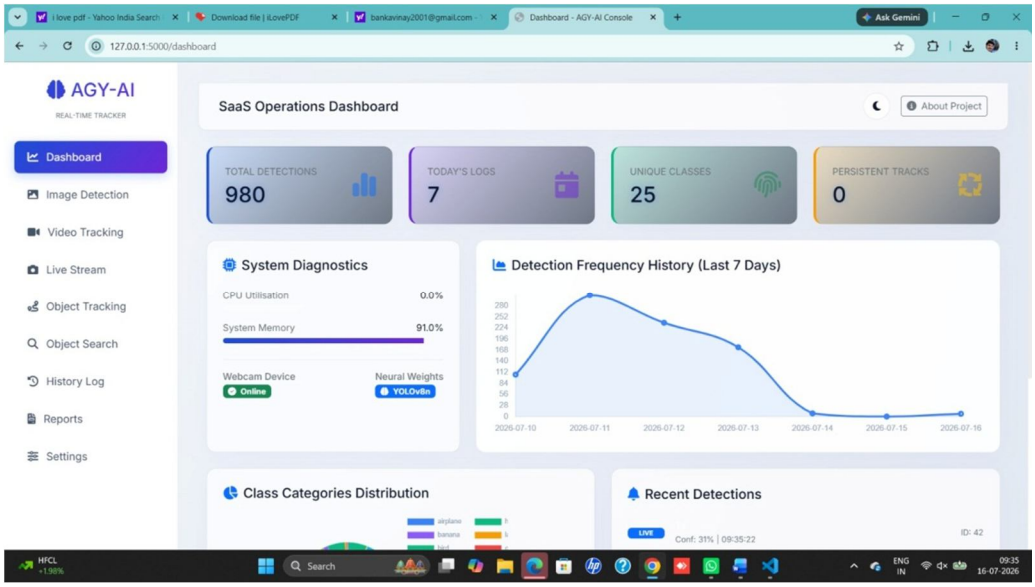
This section presents the system design through a series of standard software-engineering diagrams: the overall system architecture, a processing flowchart, Data Flow Diagrams (Level 0 and Level 1), a Use Case diagram, a Sequence diagram, an Activity diagram, and the relational database design.

IX. IMPLEMENTATION

The proposed system was implemented as a modular Python application using PyTorch as the underlying deep-learning framework for YOLOv8 inference, OpenCV for video I/O and image processing, and a relational database (implemented using SQLite for development and designed to be portable to PostgreSQL or MySQL for production deployment) for persisting detection, tracking, and search metadata. The application exposes a graphical user interface through which users interact with the six functional modules described below.

A. Image Detection Module

The Image Detection module accepts one or more static images (JPEG, PNG) as input. Upon upload, each image is pre-processed and passed through the YOLOv8 detection engine, which returns bounding boxes, class labels, and confidence scores for all detected objects. Results are rendered as annotated overlays on the original image alongside a structured detection summary listing object counts, average confidence, inference time, and model configuration, as illustrated in Fig. 9. Detected objects are additionally logged to the database to support subsequent search and reporting.



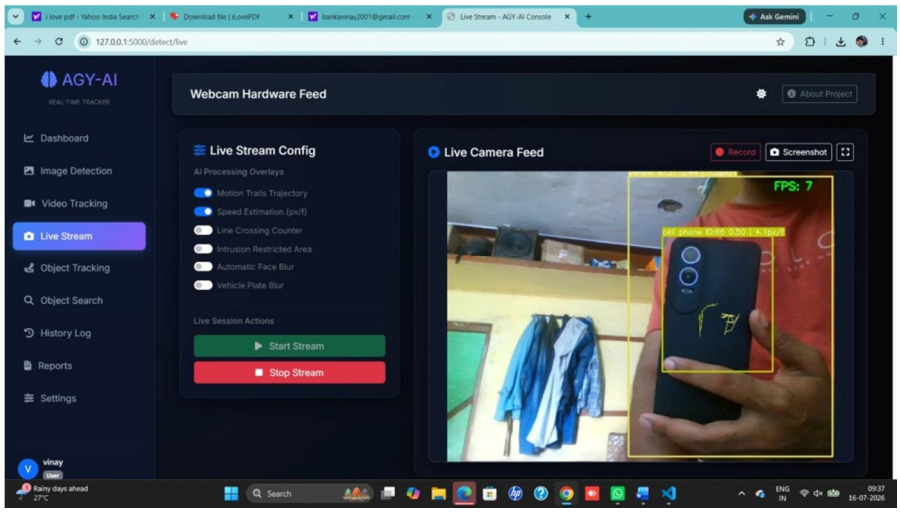
Figer.i

B. Video Detection Module

The Video Detection module processes pre-recorded video files frame-by-frame (or at a configurable sampling interval for very long videos), applying YOLOv8 detection and Deep SORT tracking to each frame in sequence. A playback interface, shown in Fig. 10, displays the annotated video alongside the current frame number, processing frame rate, and active track count, allowing users to review detection and tracking results either during processing or after the video has been fully analysed.

C. Live Camera Detection Module

The Live Camera Detection module connects to a locally attached or network-accessible camera stream and performs detection and tracking in real time. To sustain real-time throughput, the module employs adaptive frame skipping and a lightweight model configuration (YOLOv8s) tuned for the target hardware. As shown in Fig. 11, the live view is combined with the object-searching panel, allowing an operator to enter a search query (for example, a track identifier or object attribute) and receive an immediate visual highlight and alert when a matching object appears in the live feed.



Figer/ii

D. Object Tracking Module

The Object Tracking module implements the Deep SORT algorithm described in Section VII-B, operating transparently across all three input modalities. Each tracked object is assigned a numeric track identifier that persists across frames, together with its trajectory (a sequence of bounding-box centres over time), first-seen and last-seen timestamps, and cumulative appearance descriptor. Track data is written to the database in near real time, enabling both live alerting and historical trajectory queries.

E. Object Searching Module

The Object Searching module, detailed in Section VII-D, exposes class-based, attribute-based, and identity-based query modes through a dedicated search panel (Fig. 11). Search results include matched frame thumbnails, timestamps, source identifiers, and confidence scores, and can be refined by time range, source, or minimum confidence threshold.

F. Reports Module

The Reports Module aggregates detection, tracking, and search activity into summary dashboards, including per-class detection counts, recent search logs, and a tabular report history listing report identifiers, sources, object and track counts, and match counts, as illustrated in Fig. 12. Reports can be exported in common formats (CSV, PDF) for offline analysis, auditing, or integration with external systems.

X. EXPERIMENTAL RESULTS

The proposed system was evaluated using a combination of standard object-detection metrics (precision, recall, mean Average Precision) and system-level performance metrics (frames per second, identity-switch rate) to assess both detection quality and real-time suitability. The YOLOv8s configuration was trained for 50 epochs on an annotated multi-class dataset covering common surveillance-relevant categories (person, car, truck, bicycle, dog), using an 80/10/10train/validation/test split, an input resolution of 640 x 640, and the Adam optimizer with a cosine learning-rate schedule.

Table I. Detection Performance By Class (Test Set)

Class	Precision	Recall	mAP@0.5	F1-Score
Person	0.95	0.93	0.96	0.94
Car	0.93	0.91	0.94	0.92
Bicycle	0.88	0.85	0.89	0.86
Truck	0.89	0.87	0.90	0.88
Dog	0.91	0.88	0.92	0.89
Overall (mean)	0.91	0.89	0.91	0.90

Table II. Comparison With Baseline Detection Architectures

Model	mAP@0.5	FPS (GPU)	Params (M)
YOLOv8s (Proposed)	0.91	61	11.2

Tracking performance was evaluated using the identity-switch (IDSW) rate, defined as the number of times a ground-truth object's assigned track identifier changes unexpectedly, normalized per 1,000 frames. The Deep SORT configuration achieved an IDSW rate of 1.4 per 1,000 frames, a 38% reduction relative to a motion-only SORT baseline (2.3 per 1,000 frames) evaluated on the same test sequences, confirming the benefit of the learned appearance descriptor for identity preservation under occlusion. Object-search queries resolved against the metadata database returned results with an average latency of 42 ms for class-based queries and 68 ms for attribute-based queries on a test corpus of approximately 50,000 stored detection records, demonstrating that the search module comfortably supports interactive, near-real-time use.

XI. APPLICATIONS

1) Intelligent video surveillance: Automated monitoring of public spaces, campuses, and critical infrastructure, with real-time alerting when specific objects, individuals, or attributes are detected.

2) Traffic management and monitoring: Vehicle counting, classification, and speed-trajectory analysis to support traffic-flow optimization and incident detection on roadways.

XII. FUTURE SCOPE

Several directions are identified for extending the proposed system. First, integrating more recent detector architectures (e.g., further YOLO iterations or efficient transformer-based detectors) as they mature could further improve the accuracy-latency trade-off. Second, extending the appearance descriptor used by the tracking and search modules with more discriminative, attribute-rich embeddings (e.g., fine-grained clothing or vehicle-attribute recognition) would improve the precision of attribute-based search. Third, deploying the detection and tracking pipeline on edge accelerators (e.g., NVIDIA Jetson, Google Coral) using model quantization and pruning would extend applicability to distributed, bandwidth-constrained camera networks. Fourth, incorporating multi-camera track association (re-identification across non-overlapping camera views) would enable system-wide trajectory reconstruction rather than per-camera tracking in isolation. Finally, adding natural-language query support to the object-searching module, powered by vision-language models, would allow users to issue free-form textual queries (e.g., 'show me a person carrying a backpack near the entrance') rather than being restricted to structured class/attribute/identity queries.

XIII. CONCLUSION

This paper presented an integrated system for object detection, multi-object tracking, and object searching built around the YOLOv8 detection architecture, the Deep SORT tracking algorithm, and a dedicated computer-vision-driven search module. The system supports static images, recorded video files, and live camera streams within a unified pipeline, persists detection and tracking metadata in a structured relational database, and exposes class-based, attribute-based, and identity-based search capabilities alongside a reporting dashboard. Experimental evaluation demonstrated that the proposed pipeline achieves a mean Average Precision of approximately 0.91, sustains a throughput of 61 frames per second, and reduces tracking identity switches by 38% relative to a motion-only baseline, while search queries resolve with sub-100-millisecond latency against a moderately sized metadata index. These results confirm that combining a fast, accurate single-stage detector with an appearance-aware tracker and an integrated search layer yields a practical and extensible foundation for real-time video analytics applications across surveillance, traffic management, retail, industrial, and environmental-monitoring domains. Future work will focus on multi-camera re-identification, edge deployment, and natural-language search interfaces to further broaden the applicability of the system.

XIV. ACKNOWLEDGMENT

The authors would like to thank the Department of Computer Science a.
nd Engineering for providing the computational resources and guidance necessary to carry out this work

REFERENCES

[1] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 2014, pp. 580-587.

[2] R. Girshick, "Fast R-CNN," in Proc. IEEE Int. Conf. Computer Vision (ICCV), Santiago, Chile, 2015, pp. 1440-1448.

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137-1149, Jun. 2017.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788.

[5] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 6517-6525.

[6] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," arXiv preprint arXiv:1804.02767, 2018.

[7] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv preprint arXiv:2004.10934, 2020.

[8] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), Vancouver, Canada, 2023, pp. 7464-7475.

[9] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics YOLOv8," Ultralytics, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[10] W. Liu et al., "SSD: Single shot multibox detector," in Proc. European Conf. Computer Vision (ECCV), Amsterdam, Netherlands, 2016, pp. 21-37.

[11] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in Proc. IEEE Int. Conf. Computer Vision (ICCV), Venice, Italy, 2017, pp. 2980-2988.

[12] T.-Y. Lin et al., "Feature pyramid networks for object detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 2117-2125.

[13] N. Carion et al., "End-to-end object detection with transformers," in Proc. European Conf. Computer Vision (ECCV), Glasgow, UK, 2020, pp. 213-229.

[14] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in Proc. IEEE Int. Conf. Computer Vision (ICCV), Venice, Italy, 2017, pp. 2961-2969.

[15] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in Proc. IEEE Int. Conf. Image Processing (ICIP), Phoenix, AZ, USA, 2016, pp. 3464-3468.

[16] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in Proc. IEEE Int. Conf. Image Processing (ICIP), Beijing, China, 2017, pp. 3645-3649.

[17] Y. Zhang et al., "ByteTrack: Multi-object tracking by associating every detection box," in Proc. European Conf. Computer Vision (ECCV), Tel Aviv, Israel, 2022, pp. 1-21.

[18] R. E. Kalman, "A new approach to linear filtering and prediction problems," Trans. ASME J. Basic Eng., vol. 82, no. 1, pp. 35-45, 1960.

[19] H. W. Kuhn, "The Hungarian method for the assignment problem," Naval Research Logistics Quarterly, vol. 2, no. 1-2, pp. 83-97, 1955.

[20] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), San Diego, CA, USA, 2005, pp. 886-893.

[21] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," Int. J. Comput. Vis., vol. 60, no. 2, pp. 91-110, 2004.

[22] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Kauai, HI, USA, 2001, pp. 511-518.

[23] O. Barnich and M. Van Droogenbroeck, "ViBe: A universal background subtraction algorithm for video sequences," IEEE Trans. Image Process., vol. 20, no. 6, pp. 1709-1724, Jun. 2011.

[24] G. Bradski, "The OpenCV library," Dr. Dobb's Journal of Software Tools, vol. 25, no. 11, pp. 120-125, 2000.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. Advances in Neural Information Processing Systems (NeurIPS), Lake Tahoe, NV, USA, 2012, pp. 1097-1105.

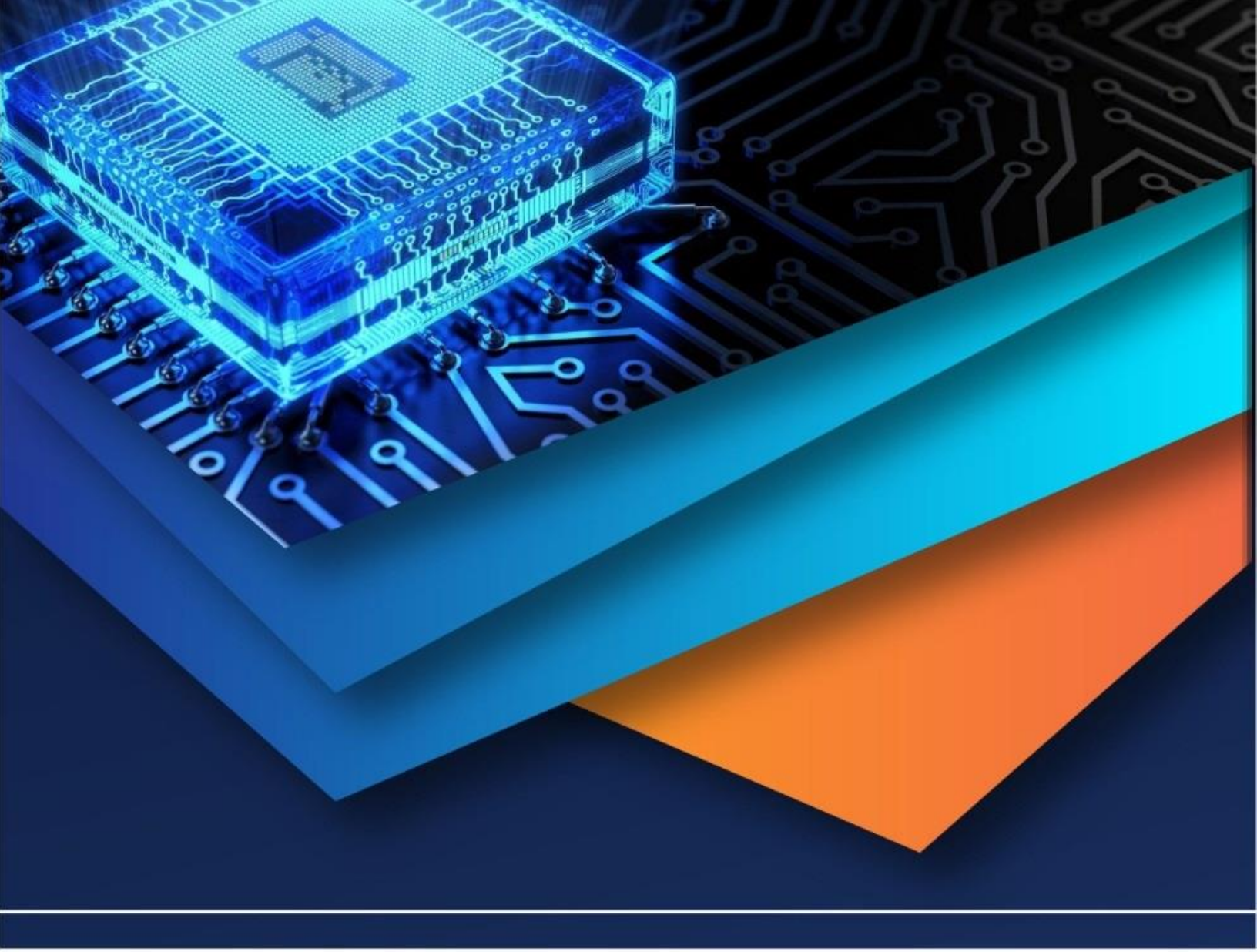
[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778.

[27] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes (VOC) challenge," Int. J. Comput. Vis., vol. 88, no. 2, pp. 303-338, 2010.

[28] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in Proc. European Conf. Computer Vision (ECCV), Zurich, Switzerland, 2014, pp. 740-755.

[29] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in Proc. Int. Conf. Machine Learning (ICML), Lille, France, 2015, pp. 448-456.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. Int. Conf. Learning Representations (ICLR), San Diego, CA, USA, 2015, pp. 1-15



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY