*International Journal of Computer Applications (0975 - 8887)*
*Volume 186 - No.66, February 2025*

# Computer Vision and Deep Learning based Approach for Traffic Violations due to Over-speeding and Wrong Direction Detection

Shailendra Singh Kathait
Co-Founder and Chief Data Scientist
Valiance Solutions
Noida, India

Ashish Kumar
Principal Data Scientist
Valiance Solutions
Noida, India

Samay Sawal
Intern Data Scientist
Valiance Solutions
Noida, India

Ram Patidar
Data Scientist
Valiance Solutions
Noida, India

Khushi Agrawal
Intern Data Scientist
Valiance Solutions
Noida, India

## ABSTRACT

Important traffic regulation and safety in High populated and fast developing urban areas. Traditional applications of traffic laws involving enforcement of speed violations or illegal directions often leverage upon pricey infrastructure, especially Automatic Number Plate Recognition cameras [14] which capture data about the vehicle and eventually process it. In the following paper there is presenta a new approach - one that is cost efficient and scalable for detect vehicle speed and wrong direction violations using publicly available non-specialized cameras. The methodology in paper uses state of art deep learning object detection models namely YOLO based architectures and advanced computer vision techniques for accurate, real time speed estimation and direction detection of vehicles. It is demonstrated how the proposed system can flag critical traffic violations such as overspeeding and traveling in the wrong direction. The system's modular design and dependency on general purpose cameras This approach must allow for its widespread, affordable implementation. Experimental results show us the robustness, accuracy, and real time capabilities of the proposed approach as well as insight into practical deployment in real world traffic surveillance applications.

## Keywords

Computer Vision, Traffic Surveillance, YOLO, Vehicle Speed Detection, Direction Detection, Non-ANPR Cameras, Frames, Centroid, Speed Estimation, Euclidean Distance, Bounding Box

## 1. INTRODUCTION

The rapid urbanization and increase of personal vehicles have introduced complex challenges in traffic management and road safety. Issues such as over-speeding, illegal parking, and driving against traffic flow are not only wrong to smooth traffic flow but also pose severe risks to people walking on footpath and motorists. Traditionally, law applying agencies have been dependent heavily on specialized infrastructure such as Automatic Number Plate Recognition (ANPR) cameras and radar guns to monitor and prosecute traffic violations. However, even though these are effective, they tend to be costly, singular and can be afforded at just a few critical checkpoints only. Besides, setting up a speed radar at every other critical junction is financially difficult besides being a logistically unfeasible solution for big coverage particularly in highly populated or dynamic growth urban centers.

The new computer vision and deep learning [9] breakthrough has opened a copy of windows on solving such problems. Availability of common public cameras which was installed for security or general surveillance purposes offers a low cost and scalable alternative. By re purposing these existing cameras traffic violations such as over speeding and wrong direction movement can be monitored at scale without high investments required for traditional technologies.

In this research, a new computer vision based framework is proposed that uses widely available public camera feeds to detect speed and wrong direction violations. By integrating state of art object de-detection models such as YOLO (You Only Look Once) with tracking algorithms and geometry based analysis, it is shown a cost-effective, scalable, and accurate solution for real time traffic surveillance.

### 1.1 Motivation

The motivation behind this work was in reducing costs, scaling application coverage, and enhancing the effectiveness of traffic law application. By converting any standard surveillance camera into a traffic monitoring sensor city government can:

(1) Expand monitoring to more locations without much hardware investments

(2) Increase the frequency and consistency of traffic law application thereby improving compliance
(3) Integrate with existing smart city solutions and data analytics platforms for comprehensive urban mobility insights

### 1.2 Contributions

The primary contributions of the work are:

(1) **Novel Use of Public Cameras**: Demonstrating that widely available, non-specialized CCTV or security cameras can be effectively repurposed for advanced traffic violation detection, negating the need for expensive ANPR or radar systems
(2) **Integrated Detection and Tracking**: Implementing YOLO based multi class object detection models to identify vehicles, motorcyclists, bicycles, and people walking on footpath, and integrating them with tracking algorithms to maintain object identities over multiple frames
(3) **Speed Estimation Methodology**: Introducing a computer vision based speed calculation method that uses frame by frame displacement of tracked vehicles, combined with appropriate calibration factors derived from geometry of frame and frame rate to estimate speed accurately
(4) **Direction Inference for Violations**: Employing a centroid based approach to see travel direction and flagging vehicles moving counter to the designated direction of traffic
(5) **Modular and Scalable Design**: Presenting a system that can be easily adapted to various camera setups and scenes, and integrated with cloud based infrastructure for remote processing and storage
(6) **Optimized Signal Timings**: If a large number of vehicles is detected while the light is red a signal control could extend or shorten the red phase accordingly to manage traffic more efficiently.
(7) **Law Application and Violation Detection**: Authorities can easily point vehicles crossing during red or yellow signals, helping in automated challans or warnings.

## 2. RELATED WORK

Conventional methods for speed application was dependent heavily on radar based speed guns and ANPR cameras to detect and identify speeding vehicles [1]. These systems while precise are relatively expensive and limited to specific checkpoints. With the rise of deep learning based object detectors, several studies have explored detecting vehicles and estimating speed using general camera feeds [2][3]. However many of these approaches require extensive calibration and specialized hardware.

YOLO and other deep learning architectures like Faster R-CNN and SSD have largely advanced real time object detection in surveillance footage [4]. Integrating detection with multi object tracking (MOT) algorithms has proven effective for long-term behavior analysis of detected objects in video streams [5]. More recent studies have begun exploring the integration of additional functionalities like direction detection [6] and violation detection [7]. However, few have thoroughly integrated speed and direction estimations in a single framework while focusing specifically on using general purpose public cameras for cost effectiveness and scalability.

The work in this paper stands on the shoulders of these advancements, using high-performance YOLO-based detectors with robust tracking and geometric analysis for speed and direction inference, while emphasizing affordability and scalability through the use of existing public camera networks.

## 3. TRAFFIC SIGNAL DETECTION AND VEHICLE DENSITY ANALYSIS

For the rapidly expanding urban centers, efficient management of traffic is the prime requirement. It requires strict adherence to the traffic signals-red, yellow, and green-for ensuring safety and reducing congestion. The difficulty lies in human error as well as its practical impossibility of checking every crossing. This task can be easily executed using computer vision and deep learning, as they give quite robust solutions for monitoring the states of signal identification in real-time vehicle behavior. This enables:

(1) Efficient resource allocation for traffic police and application agencies.
(2) Data-driven insights into congestion levels and violation trends.
(3) Enhanced safety by quickly detecting violators and reducing signal related road incidents.

In this paper, two key components have been focused on:

(1) Signal state detection via ROI brightness analysis.
(2) Vehicle counting using YOLO-based object detection models during red or yellow phases.

### 3.1 Signal State Detection

(1) **Region of Interest (ROI) Selection** Predefined bounding boxes (or ROIs) are drawn around each of the three traffic lights (red, yellow, and green). Specifically, for each traffic light, the store coordinates $(x_1, y_1, x_2, y_2)$ that allow us to extract a small region from the full image frame.
(2) **Brightness Calculation** Once the ROI is selected, the cropped image patch is converted to grayscale. The average brightness $\beta$ of the ROI is calculated by summing the intensities of all pixels and dividing by the total number of pixels:

$$\beta = \frac{\sum_{i=1}^{N} I(i)}{N},$$

where $I(i)$ is the grayscale intensity of the $i$-th pixel, and $N$ is the total number of pixels within the ROI.
(3) **Threshold Determination** For each light (red, yellow, green), sample frames are collected to compute a baseline brightness value. A brightness threshold $\beta_{th}$ is then defined (e.g., mean brightness minus a small offset) to account for lighting variations. During actual inference, if $\beta > \beta_{th}$ in the corresponding ROI, it is inferred that the light is *ON*.

### 3.2 Vehicle Detection and Counting

(1) **Object Detection** The research utilizes a YOLO-based model (e.g., YOLOv8) to detect vehicles. For each frame, the detector provides bounding boxes, confidence scores, and class IDs (e.g., car, motorcycle, bus, truck).
(2) **Vehicle Tracking** To maintain consistent identities across frames, a tracking mechanism has been employed. A dictionary or track manager stores the positions and identifiers (IDs) of vehicles in each frame. When a detection is matched to an existing track, it retains the same ID; otherwise, a new ID is assigned.

(3) **Counting Vehicles During Red or Yellow Light** Whenever the brightness check indicates that the traffic light is red or yellow, the system activates a vehicle-counting procedure. Each newly detected vehicle (not previously counted for that signal phase) is added to the count. This information is especially useful for quantifying congestion at intersections during non-green signals.

(4) **Maintaining the Vehicle Count** All vehicle counts, along with their timestamps or frame numbers, are stored for subsequent analysis. These counts:
   (a) Reflect the density or buildup of vehicles at red or yellow signals.
   (b) Can be compared against green-signal counts to study traffic flow dynamics.
   (c) Provide valuable input to dynamic signal controllers or for evaluating the effectiveness of traffic policies.

Whenever the signal turns green; model starts to detect the speed and direction of the vehicle, the methodology of which is discussed below. While the signal is red, taking a left turn is assumed to be legal by the laws as well and thus they are not flagged for wrong direction.

## 4. METHODOLOGY FOR SPEED AND WRONG DIRECTION DETECTION

The proposed framework comprises several key components: data acquisition, object detection, multi-object tracking, speed estimation, direction inference, and violation flagging. Figure 1 provides an overview of the pipeline.

### 4.1 Data Acquisition

It is assumed that access to publicly available camera feeds—standard CCTV cameras installed at junctions, parking areas, or along highways are available. These cameras are typically low to medium resolution and may not provide dedicated calibration information. The approach accounts for these constraints by employing a polygon-based scene understanding step

Thus no specialized dataset was required. The method was tested on arbitrary public domain city junction footages. The YOLO models were pretrained on popular object detection datasets (e.g., MS COCO) and fine-tuned with some domain-specific images if available.

### 4.2 YOLO Model Architecture

The YOLO (You Only Look Once) architecture has revolutionized object detection by offering real-time, highly accurate detection capabilities. Unlike traditional object detection methods that require multiple stages of processing, YOLO operates on a single neural network, combining object classification and localization into a unified framework.

Unlike traditional methods that involve a multi-stage pipeline (e.g., region proposal generation followed by classification), YOLO processes an entire image in a single neural network forward pass. This end-to-end design enables real-time detection while maintaining accuracy. Below, the paper delves into the key components and processes that define how YOLO works.

(1) **Residual Blocks:** YOLO begins by dividing the input image into a grid of N × N cells. Each cell is responsible for predicting the bounding box and the class probabilities of the objects it encompasses. This grid-based mechanism simplifies object localization by assigning distinct responsibility to each cell, ensuring that overlapping object regions are managed efficiently.
   (a) **Image Preprocessing**: The input image is resized to a fixed dimension, typically 448 × 448, ensuring compatibility with the network architecture
   (b) **Grid Division**: The image is divided into equal N × N grid cells, where each cell can predict multiple bounding boxes and their associated confidence scores
   (c) **Residual Connections**: To improve gradient flow and enable deeper network structures, YOLO employs residual blocks. These connections help YOLO retain spatial information from earlier layers and stabilize training

(2) **Bounding Box Regression:** For every grid cell, YOLO predicts bounding boxes that encompass objects within the cell's area of responsibility. Each bounding box prediction includes:

   (a) $P_c$ : The confidence score indicating the likelihood of an object being present in the bounding box
   (b) $(b_x, b_y)$: The x and y coordinates of the bounding box center, normalized with respect to the cell's grid dimensions
   (c) $(b_w, b_h)$: The width and height of the bounding box, normalized with respect to the entire image dimensions.
   (d) **Class Probabilities:** The probabilities of each object class being present in the bounding box

The overall prediction is represented as a vector:

$$Y = [p_c, b_x, b_y, b_w, b_h, c_1, c_2, \ldots, c_C]$$

Where C is the number of object classes. This unified vector ensures simultaneous localization and classification for every bounding box

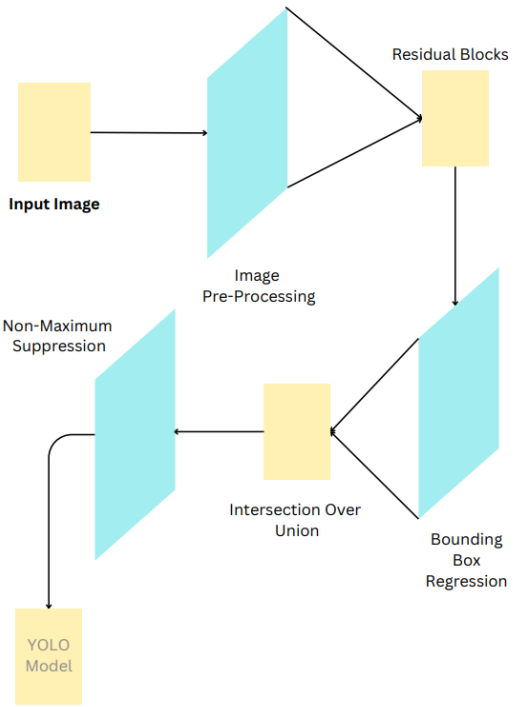


Fig 1. YOLO Object Detection Model Architecture

(3) **Intersection Over Union (IOU):** The challenge in object detection lies in eliminating redundant predictions for the same object. YOLO employs the Intersection Over Union (IOU) metric to refine predictions:
   (a) **IOU Definition:** IOU measures the overlap between the predicted bounding box and the ground truth bounding box. Mathematically, it is defined as:

$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

   (b) **Thresholding**: During training, bounding boxes with an IOU below a predefined threshold (e.g., 0.5) are discarded. This ensures that only predictions closely aligned with the ground truth are retained
   (c) **Grid Selection**: For each grid cell, only the bounding boxes with IOU greater than threshold are considered significant. This prevents false positives from influencing the training process

(4) **Non-Maximum Suppression (NMS):** While the IOU mechanism eliminates some redundant predictions, multiple high-confidence bounding boxes may still overlap for the same object. Non-Maximum Suppression (NMS) further filters these bounding boxes to retain the most relevant ones:
   (a) **Ranking**: YOLO ranks all bounding boxes based on their confidence scores
   (b) **Suppression**: For each object class, NMS iteratively removes bounding boxes with a high overlap (IOU greater than threshold) but lower confidence than the highest-ranked box
   (c) **Result:** After applying NMS, YOLO retains only the bounding box with the highest detection score for each object, reducing noise and improving prediction quality

(5) **Activation and Optimization:**
   Activation Functions:
   (a) **ReLU [13]**: Used in all layers except the final layer, facilitating efficient gradient flow and convergence.
   (b) **Linear Activation:** Applied in the final layer for bounding box coordinates to allow unrestricted values.
   Regularization:
   (a) **Batch Normalization:** Reduces internal covariate shifts, stabilizing and accelerating training.
   (b) **Dropout**: Mitigates overfitting by randomly deactivating neurons during training

The YOLO architecture stands out due to its simplicity and efficiency, combining feature extraction, classification, and localization into a single network. By leveraging residual blocks, bounding box regression, IOU, and NMS, YOLO achieves real-time object detection with competitive accuracy, making it ideal for applications requiring speed and scalability. Below is the figure for YOLO Architecture for Object Detection:

### 4.3 Model Architecture and Discussion

YOLO was used as the feature extractor here. It is a real-time object detection system that processes an image in a single neural network pass, making it both fast as well as efficient for tasks like object detection and tracking.

Step-wise YOLO Model Process:

Input Image → Grid Based Detection → Bounding Box Prediction → Class Prediction → Non Maximum Suppression

The algorithm utilizes the YOLO Model to identify and localize the vehicles. It assigns and maintains consistent tracking IDs for vehicles across different frames. There are various components dependant on Computer Vision for speed tracking and direction detection.
For each detection YOLO provides a bounding box, defining position of the vehicle in image,a class label indicating the type of vehicle detected, a confidence score indicating the model's certainty in the detection

(1) **Object Tracking:** To maintain consistent identities for each detected vehicle, a built in tracker is integrated that associates detections across consecutive frames. This tracker relies on comparing bounding box features such as centroids and sizes between the current and previous frames. Persistent object IDs ensure that speed and direction can be computed over time.

(2) **Centroid Calculation:** Once a vehicle is detected, centroids are computed:

$$\text{centroid}_x = \frac{x_1 + x_2}{2}, \quad \text{centroid}_y = \frac{y_1 + y_2}{2}.$$

   This geometric center provides a simplified representation of the vehicle's position and is less sensitive to bounding box size variations.
   In each frame centroids are stored [10] calculated in that frame before moving on to the next frame and finding centroids again based on the bounding box coordinates and coordinates for the centroid for a particular vehicle in certain frame is found using the ID of the vehicle and the vehicle region.
   Because the tracking IDs are persistent across frames, and because each frame updates the stored centroid position, the code effectively creates a timeline of centroid positions.

   As each new frame arrives:

   (a) looking up at the existing tracking IDs.
   (b) Retrieve the previous frame's centroid.
   (c) Calculate differences in position to infer speed, direction, or other metrics.
   (d) Update the dictionaries with the most recent centroid and discards older entries once they're no longer needed.

   Thus, successive centroid tracking is accomplished by combining YOLO's built-in object tracking (which provides stable track IDs per object across frames) with dictionaries that store and update each object's centroid from one frame to the next. By referencing these stored positions, changes in position can be determined over time, thus analyzing movement patterns such as speed, direction, or violations.

   Below is an example of the frames which showcase tracking of centroid of a vehicle (3-wheeler):


Fig 2: Red Bounding Box for violation due to Over-speeding Frame 1 (Red Cross - Centroid)


Fig 3: Red Bounding Box for violation due to Over-speeding Frame 2 (Red Cross - Centroid)

The above two frames showcase the centroid in the bounding box and how the centroid is tracked and maintained in between the two frames.

(3) **Speed Estimation Process:** Speed [11] calculation is central to the system's functionality. A inter-frame differences is leveraged to estimate velocity:

(a) **Position Logging:** For each tracked vehicle, centroid are stored and the frame number in a dictionary keyed by track ID.

(b) **Distance and Time Computation:** When a vehicle reappears in a subsequent frame, previous position are retrieved $(x_{\text{prev}}, y_{\text{prev}}, \text{frame}_{\text{prev}})$. The Euclidean distance [12] traveled is:

$$d = \sqrt{(x_{\text{current}} - x_{\text{prev}})^2 + (y_{\text{current}} - y_{\text{prev}})^2}.$$

The time elapsed is computed from the difference in frame indices:

$$\Delta t = \frac{\text{frame}_{\text{current}} - \text{frame}_{\text{prev}}}{\text{frame\_rate}}.$$

(c) **Speed Conversion:** Initially, speed in pixels per second is derived:

$$v_{\text{pixels/s}} = \frac{d}{\Delta t}.$$

A known scale factor, derived from scene calibration, converts pixel-based measurements to real-world units (e.g., km/h):

$$v_{\text{km/h}} = v_{\text{pixels/s}} \times \text{speed\_multiplication\_factor}.$$

Minor movements below a threshold are rounded down to zero to reduce noise.

(4) **Direction Compliance Check:** To determine whether a vehicle moves in the correct or opposite direction, polygonal regions are defined with expected directions. Each region is associated with a direction: for instance, downward $(+1)$ or upward $(-1)$ travel in image coordinates. the last several (e.g., six) centroid positions are stored of each vehicle within a given region. After accumulating these positions:

(a) Compute the displacement in the vertical direction:

$$\Delta y = y_{\text{newest}} - y_{\text{oldest}}.$$

(b) Determine the actual direction:
   i. If $\Delta y > 0$, the vehicle is moving downward.
   ii. If $\Delta y < 0$, the vehicle is moving upward.

(c) Compare the inferred direction with the expected direction of the region:
   i. If the expected direction is downward and the vehicle moves upward, it violates direction rules.
   ii. If the expected direction is upward and the vehicle moves downward, it also constitutes a violation.

When a violation is detected, the system highlights the bounding box in red and increments a violation counter.

To ensure accurate detection of traffic violations, it is crucial to define a Region of Interest (ROI) that focuses on the relevant sections of the road. This allows the system to exclude unnecessary background and concentrate on the areas where vehicles are expected to move. If both sides of the lanes are visible, separate logic is applied for each direction of traffic, as vehicles in opposing lanes may have different directional rules. For instance, upward-moving vehicles in one lane could indicate a violation, while the same motion in the opposite lane is valid. This distinction ensures precise identification of both speed and directional violations for multi-lane scenarios.

### 4.4 Visualization and Output:

For each processed frame, the system overlays:

(1) Bounding boxes around detected vehicles
(2) Labels including vehicle type and track ID
(3) Computed speed text near the bounding box
(4) Directional compliance messages and color-coded bounding boxes in the event of a violation

This immediate feedback facilitates operator monitoring and provides an annotated video output for analysis. Below are the output of the frames and bounding box overlayed to flag over-speeding and Opposite direction violations:


Fig 4. Red Bounding Box to flag over-speeding vehicles


Fig 5. Red Bounding Box to flag over-speeding vehicles


Fig 6. Blue Bounding Box to flag vehicles moving in wrong direction


Fig 7. Blue Bounding Box to flag vehicles moving in wrong direction


Fig 8. Blue Bounding Box to flag vehicles moving in wrong direction

## 5. CONCLUSION AND FUTURE WORK

This paper presented a deterministic, novel, low-cost, and flexible solution for detecting key traffic violations using readily available public camera feeds. By leveraging advanced YOLO-based object detection models, robust multi-object tracking, and polygon-based scene understanding, the research also demonstrated a system capable of identifying speed and directional violations, as well as other infractions like helmet non-compliance and cycle lane misuse. This paper also included a novel approach to detect signal and execute methods based on the color of the signal

The results show that such a system can provide real-time feedback to traffic authorities, significantly reducing the need for expensive specialized hardware. With future refinements and scalability improvements, this approach promises a transformative impact on urban traffic management, safety monitoring, and law enforcement.

Future improvements could include:

(1) **Integrating camera calibration techniques to enhance speed estimation accuracy:** Integrating camera calibration techniques is crucial to reducing measurement errors in speed estimation. Calibration involves determining the camera's intrinsic parameters (like focal length and lens distortion) and extrinsic parameters (such as its orientation and position relative to the scene). By accurately mapping image coordinates to real-world dimensions, the system can more precisely calculate vehicle speeds. This not only minimizes distortions caused by perspective but also adapts to environmental changes over time, ensuring consistent performance

(2) **Employing advanced models for nighttime detection and infrared camera feeds:** Traditional detection models often struggle under low-light or nighttime conditions due to reduced visibility and increased noise. By employing advanced models designed for these scenarios, including those optimized for infrared (IR) camera feeds, the system can maintain high detection accuracy regardless of ambient light. Infrared cameras capture thermal signatures, providing a reliable alternative to visible-light imaging during the night. These improvements can lead to a more robust detection system that effectively tracks vehicles in all lighting conditions

(3) **Incorporating ANPR (if available) for vehicle identification and integrating directly with traffic databases:** Incorporating Automatic Number Plate Recognition (ANPR) technology enables the system to automatically extract license plate information from detected vehicles. This unique identifier can then be cross-referenced with traffic databases, allowing for real-time vehicle identification, history tracking, and integration with broader traffic management systems. This feature is particularly valuable for law enforcement, tolling, and congestion management, as it facilitates swift data retrieval and decision-making, reducing manual intervention and enhancing overall traffic surveillance

(4) **Utilizing vehicle re-identification models and data fusion from multiple camera angles for comprehensive city-wide tracking:** Vehicle re-identification (Re-ID) models are designed to recognize and track the same vehicle across different camera views, even if the vehicle's appearance changes due to lighting, angle, or occlusion. By fusing data from multiple camera angles across a city, the system can construct a comprehensive picture of traffic flow and vehicle movement. This holistic tracking is essential for analyzing congestion patterns, planning urban infrastructure, and enabling rapid responses in emergency situations. Data fusion from various sources not only enhances tracking accuracy but also enriches the dataset for better analytics and predictive modeling

## 6. REFERENCES

[1] S. Sutanto et al., "Real-Time Vehicle Speed Estimation and License Plate Recognition Using Intelligent Transportation Systems," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 5, pp. 2192–2202, 2020

[2] Y. Wang, E. K. Teoh, and D. Shen, "Lane detection and tracking using B-Snake," Image and Vision Computing, vol. 22, no. 4, pp. 269–280, 2004

[3] S. Cho et al., "Vision-based vehicle detection system with speed estimation," Optical Engineering, vol. 50, no. 12, 2011

[4] J. Redmon et al., "You Only Look Once: Unified, Real-Time Object Detection," CVPR, 2016

[5] TA. Bewley et al., "Simple Online and Realtime Tracking," ICIP, 2016

[6] Z. Fang et al., "Robust vehicle detection in challenging urban scenarios," IEEE Intelligent Vehicles Symposium, 2018

[7] L. Sun et al., "Helmet Detection on Motorcyclists Using Deep Learning," International Journal of Computer Vision, 2020

[8] Y. Xiong, J. Ma, and J. Zhou, "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," arXiv preprint arXiv:2110.06864, 2021

[9] Kartikya Gupta, Vaibhav Sharma, Shailendra Singh Kathait (2024) Valiance Analytics Private Limited. Smart Screening: Non-Invasive Detection of Severe Neonatal Jaundice using Computer Vision and Deep Learning. International Journal of Computer Applications (0975 – 8887) Volume 186 – No.35, August 2024

[10] Rod Deakin, S. C. Bird, R. I. Grenfell December 2002, The Centroid? Where would you like it to be be?, Cartography 31(2):153-167, DOI: 10.1080/00690805.2002.9714213

[11] Danang Wicaksono, Budi Setiyono March 2017, Speed Estimation On Moving Vehicle Based On Digital Image Processing, International Journal of Computing Science and Applied Mathematics 3(1):21, DOI: 10.12962/j24775401.v3i1.2117

[12] Leo Liberti, Carlile Lavor, Nelson Maculan, Antonio Mucherino May 2012, Euclidean Distance Geometry and Applications, SIAM Review 56(1), DOI:10.1137/120875909

[13] Abien Fred Agarap, March 2018, Deep Learning using Rectified Linear Units (ReLU), arXiv:1803.08375 [cs.NE], https://doi.org/10.48550/arXiv.1803.08375

[14] Chirag Indravadanbhai Patel, D. Shah, Atul Patel, May 2013, Automatic Number Plate Recognition System (ANPR): A Survey, International Journal of Computer Applications, 69(9):21-33, DOI: 10.5120/11871-7665

[15] Shailendra Singh Kathait, Sakshi Mathur Valiance Analytics Private Limited. Machine-learning based Hybrid Method for Surface Defect Detection and Categorization in PU Foam. International Journal of Computer Applications (0975 - 8887) Volume 181 - No.25, November 2018