Mustansiriyah Journal of Pure and Applied Sciences, Print ISSN: 2957-9910. Online ISSN: 2957-9929, Vol. 4, No. 1, 2026, Pages 50-66
DOI: 10.47831/mjpas.v4i1.335

MJPAS

MUSTANSIRIYAH JOURNAL OF PURE AND APPLIED SCIENCES

Journal homepage:
https://mjpas.uomustansiriyah.edu.iq/index.php/mjpas

RESEARCH ARTICLE – COMPUTER SCIENCE

Vehicle Speed Estimation Using Faster RCNN

Salih Khalid Abdulmonem [1*], Anwar H. Al-Saleh [2], Intisar Abd Yousif [3]

[1,3] Department of Computer Science, College of Education, Mustansiriyah University, Iraq
[2]Department of Computer Science, College of Science, Mustansiriyah University, Iraq
[*] Corresponding author E-mail: salikhalid@uomustnsiriyah.edu.iq

Article Info.

Article history:

Received
23 October 2024

Accepted
26 December 2024

Publishing
30 March 2026

Abstract

The transport network, especially the road system, has faced huge pressures due to increased population growth and urbanization. One of the negative side effects of traffic growth is road accidents. Intelligent Transportation system (ITS) is like the Human vision system (HVS). Deep learning-based classification and detection algorithms have emerged as powerful tools for vehicle detection in intelligent transportation systems. The limitation of the number of high-quality labeled training samples makes the single vehicle detection in intelligent transportation systems. This paper presents the detection and tracking of vehicles on the recorded dataset by utilizing the Faster RCNN architecture and Matlab 2023 programming language. After training the object detection model using Faster RCNN the detection accuracy 100% and then applying Kalman Filter to track the vehicles detected by the Faster RCNN. The system has proven its effectiveness in detecting and distinguishing vehicles.

The official journal published by the College of Education at Mustansiriyah University

Keywords: vehicle detection, machine learning, Faster RCNN, intelligent transport system, Kalman filter tracking.

Introduction

We create a system for vehicle identification, tracking, and speed estimate since the streets of Baghdad have become very congested as a result of this issue. For many years, vehicle identification techniques have been expanding in both academia and industry. To yet, the majority of cutting-edge object identification techniques have not been able to match vehicle detection standards. The main obstacles in vehicle detection include wide fluctuations in light, extensive occlusion, and big differences in object sizes. The challenge of vehicle recognition is for a computer to identify a car in the same manner that a human does. The government often uses vehicle identification for things like computerized traffic tickets and road upkeep. An picture of a car may contain information likeclass, type, and so forth. Even if a car in the same class could have the same shape, its species may be different. With the development of vehicle identification technology, we expect that computers will be able to recognize the kind of car we are looking at, much as humans do instinctively. Road maintenance scheduling, traffic monitoring, and traffic violation warnings are just a few of the applications that vehicle categorization may now be utilized for. Identification might be challenging because the pictures are exactly like those of an automobile. Photographs of vehicles may be shot in a variety of settings, such as lighting and point-of-view. There may be a variety of images showing the car's "personal" characteristics, including its form, front end, tail, and so on.[1].

One of the most important tasks in computer vision systems is object detection. One of the most researched subjects among computer vision researchers is object detection because of its many uses, such as vehicle detection. Modern object detection algorithms are far faster and better than their predecessors because to the development of deep learning algorithms and the availability of powerful computers. Prior to 2014, practically all algorithms approached object recognition tasks using conventional computer vision ideas. The Scale Invariant Feature Transform (SIFT) method was among the first to provide a respectable level of accuracy at a

fast speed. Later, a sliding window method was used to construct the Viola-Jones Detectors (VJ Det.) algorithm.[2].

The regression function in Faster RCNN is a pyramid scheme consisting of many anchor boxes. Translation invariance and numerous size (and aspect ratio) representations are two characteristics of these anchor boxes that make them appropriate for object recognition and categorization. After that, training is conducted alternately using the RPN and Fast RCNN networks. Therefore, the entire system may be trained as a single entity end-to-end with Faster RCNN, which eliminates all of the extra processing needed by the selective search technique to create region recommendations. A linear dynamic system's state may be predicted from observations using the efficient Kalman filter. The current state estimate is calculated recursively, requiring accurate measurement data and state-space estimation just once. The structure of the EKF and the linear Kalman filter is nearly identical. The state transition matrix and linearization processes are the only differences in the measurement matrix. It is necessary to use process and measurement models to update the mean and covariance in the nonlinear Kalman filter. This transformation's mean and covariance calculations demonstrate the applicability of Kalman update equations.[3].

In the rapidly changing world of modern transportation, Intelligent Transportation Systems (ITSs) have become a disruptive force that is radically altering the way that people and goods are transported. They enhance the efficiency, security, and environmental friendliness of transportation networks by fusing state-of-the-art technology, data analytics, and communication systems. Modern information and communication technology is used by ITS to improve public transportation networks, vehicle operation, and traffic management, among other aspects of transportation. ITS uses real-time data, sensor networks, and cognitive algorithms to reduce traffic congestion, shorten travel times, increase safety, and minimize environmental impacts. Smart traffic signal systems, autonomous vehicles, dynamic route planning, electronic toll collection, and real-time public transportation tracking are just a few of the many applications that fall within the broad purview of Intelligent Transportation Systems (ITS). These developments support broader societal goals including energy conservation, lower emissions, and better urban planning in addition to improving people's daily commuting. ITS is a key factor in defining the direction of mobility in the future because to the continuous acceleration of urbanization and the rising need for effective transportation. In order to create more intelligent, connected, and ecologically friendly transportation ecosystems, governments, businesses, and researchers from all over the world are investing resources in the implementation of ITS technology. The use of ITS has accelerated in recent years due to the introduction of technologies like 5 G connectivity and the Internet of Things (IoT). Prominent ITS applications include Mobility Prediction, Vehicular Ad-hoc Networks (VANETs), Intelligent Traffic Lights (ITL), and Virtual Traffic Lights (VTL). These are meant to lessen congestion and enhance traffic flow. [4].

Deep learning-based object detection algorithms are now more developed and have the potential to perform better in a variety of applications, including intelligent transportation systems, intelligent monitoring systems, medical object identification, and military object detection. For instance, there are issues with selecting and locating the item from the limited feature information when its scale is too tiny, as well as with precisely and rapidly capturing it using only a portion of the feature information when it is partially obscured. The issue of creating better anchors for object detection is also worth bringing up. In object identification, an anchor is frequently used as a reference frame for regression and classification. The majority of widely used anchoring techniques are unable to adequately cover the object area Because of their fixed design; the majority of widely utilized anchoring techniques are unable to adequately cover the object area. An extremely high number of anchors is needed to

guarantee a high recall rate, which will lead to an excessive number of negative samples in the range that the anchors encompass. Such issues will plague the overall anchor creation approach, leading to subpar detection effectiveness. This research suggests an enhanced technique based on a quicker region-based convolutional neural network (R-CNN) with better identification performance to address the aforementioned issues. The region proposal network (RPN), selective search, and other candidate region techniques are replaced by the guided anchor approach. Based on this, the network module to address the issue is also obtained using the context feature of occlusion of objects. In order to maximize the detection performance of Faster R-CNN in complicated scenarios, the skip pooling approach is utilized for multi-feature fusion of several deep neural network methods to address the issue of tiny object scale [5].

2. Related Work

Mahmoud Famouri, Zohreh Azimifar, et. al. 2017, present a novel motion plane-based approach for vehicle speed estimation, utilizing the center of a vehicle's license plate as a reference point. By estimating a hypothetical motion plane and mapping plate positions, the method mitigates parallax displacement effects. Experimental results demonstrate superior performance compared to existing methods, highlighting the efficacy of this approach for accurate and reliable vehicle speed estimation without the need for camera calibration (Shape-from-Template (SfT))[6].

KWANG-JU KIM , PYONG-KUN KIM et al 2019, they presents a multi-scale vehicle detection network that enhances the Yolo-v3 framework by incorporating additional object prediction layers and spatial pyramid pooling (SPP) to address challenges in traffic surveillance, such as scale variations and occlusions. The proposed method significantly improves detection accuracy, achieving a mean Average Precision (mAP) of 85.29% on the UA-DETRAC benchmark dataset, particularly in crowded conditions. The results indicate that the modified architecture outperforms existing detection methods, showcasing its effectiveness for real-time traffic monitoring applications [7].

Salvatore Trubia, Alessandro Severino et al 2020, discuss the significance of Smart Roads in the context of autonomous vehicles and intelligent transportation systems. It highlights the role of V2I communication and innovative technologies in enhancing traffic management and safety. The future direction involves developing eco-friendly infrastructure monitoring systems and smart mobility contexts to optimize traffic operations and user safety (backpropagation algorithm implemented within an artificial neural network (ANN) model)[8].

Zheng Tang, Gaoang Wang et. al., present a novel framework that combines visual and semantic features for Single Camera Tracking (SCT) and Inter-Camera Tracking (ICT) of vehicles. The approach includes an adaptive appearance model, loss functions for data association, and an evolutionary algorithm for camera parameter optimization. Evaluation of the NVIDIA AI City Dataset demonstrates superior performance in 3D speed estimation and vehicle re-identification (YOLOv2) [9].

D. Sudha and J. Priyadarshini 2020, discuss various existing methods and proposed solutions for multiple vehicle detection and tracking in real-time traffic scenarios. they highlight the use of hybrid classifiers, machine learning, and modified algorithms to improve accuracy rates. The study emphasizes the importance of advanced technologies for road safety and

addresses challenges such as illumination changes, occlusion, and collision in vehicle detection and tracking systems [10].

Jiaxing Zhang, Wen Xiao et al 2020, present a vehicle tracking framework using roadside lidar data for accurate speed estimation. The framework includes vehicle detection from point clouds, centroid-based tracking, and a refinement module for speed accuracy improvement. Results demonstrate lidar sensors' capability for vehicle tracking and speed monitoring in urban environments, highlighting the potential for future traffic monitoring applications (a combination of the Unscented Kalman Filter (UKF) for initial tracking and the Joint Probabilistic Data Association Filter (JPDAF) for assignment in the tracking algorithm)[11].

Narina Thakur, Preeti Nagrath et al 2021, They presents a study on pedestrian detection in crowded scenarios using deep learning techniques, specifically focusing on the YOLO architecture. The research utilizes the state-of-the-art MOT20 dataset and custom videos captured by a drone to evaluate the effectiveness of various object detection models. Results indicate that YOLOv5 achieves superior performance, highlighting its potential for real-time applications in surveillance and crime prevention.[12]

R eddy A lexandro H arianto, Y uliana M elita Pranoto et al 2021, this paper presents a method to improve vehicle detection and recognition using Faster R-CNN combined with advanced data augmentation techniques. The study addresses common challenges such as limited training samples and uneven class distribution by employing various augmentation strategies, including image enhancement and perspective correction. Results demonstrate that these methods significantly enhance detection accuracy, providing a robust solution for real-time vehicle identification in diverse conditions [13]

Romany F. Mansour, José Escorcia-Gutierrez et al 2021, they presents an Intelligent Video Anomaly Detection and Classification (IVADC-FDRL) model that integrates Faster R-CNN with Deep Reinforcement Learning (DRL) techniques to enhance video surveillance capabilities. The model aims to automatically identify and classify abnormal events in video streams, significantly improving public security measures. Experimental results demonstrate the model's effectiveness, achieving accuracies of 98.50% and 94.80% on two distinct datasets, showcasing its potential for real-time anomaly detection in complex environments [14].

Refaat Mohammad Alamgir, Ali Abir Shuvro et al 2022, they presents a comprehensive evaluation of various YOLO-based vehicle detection algorithms, specifically YOLOv3, YOLOv5s, and YOLOv5x, utilizing a curated dataset primarily sourced from DhakaAI. The study focuses on the trade-off between accuracy and speed, essential for real-time applications in intelligent transport systems. Results indicate that YOLOv5x achieves the highest mean Average Precision (mAP) of 28.7%, while also addressing challenges such as class imbalance and occlusions, with plans for future enhancements to the dataset and exploration of emerging YOLO variants [15].

Prashan Premaratne, Inas Jawad Kadhim et al 2023, they presents a comprehensive review of vehicle detection, classification, and counting on highways, emphasizing the advancements in computer vision techniques, particularly through deep learning and neural networks. The study highlights the effectiveness of YOLOv5 models in achieving high accuracy for real-time vehicle counting, while also addressing challenges such as low light conditions that affect detection performance. The findings suggest that integrating traditional image processing methods with modern deep learning approaches can significantly enhance the accuracy of vehicle monitoring systems.(YOLOv5l achieved 99.2% accuracy)[16]

H.J. Kadhim, A.H. Abbas 2024, they examines the advancements in autonomous vehicles (AVs) focusing on safety measures and the integration of artificial intelligence. It reviews essential technologies such as computer vision, sensor fusion, and vehicular communication systems, highlighting the need for comprehensive assessment methodologies. The study identifies current challenges in AV design and implementation, including cybersecurity threats and environmental factors, while proposing future research directions to enhance the safety and efficiency of self-driving cars [17].

Zahraa Haimeed Rasool, Maha Adham Abdel Amir 2024, they presents a hybrid CNN-LSTM algorithm designed for object detection within the COCO dataset, encompassing various everyday objects, animals, and vehicles. The approach integrates advanced preprocessing techniques, including grayscale conversion, histogram equalization, and median blur for noise reduction, followed by robust feature extraction methods such as Principal Component Analysis (PCA), Gray-Level Co-Occurrence Matrix (GLCM), and Histogram of Oriented Gradients (HOG). The proposed model achieves high classification accuracy, demonstrating its effectiveness in enhancing object detection capabilities in computer vision applications, the accuracy of the proposed hybrid CNN-LSTM model is 0.9917 [18].

One of the most fundamental and critical tasks in computer vision is to detect moving objects along with their video sequence movement. This provides the basis for a variety of automated applications in a range of domains, including monitoring, increased reality and movement capture. Object monitoring is a key component of the IVS or Intelligent Video Surveillance program and can further is replicated for other suspicious activity detection systems.

The goal of this paper is to build a system for vehicle detection and tracking in Baghdad city, the dataset used in training and testing was recorded by our camera in Baghdad Iraq; we use Faster RCNN with modified layers for vehicle detection and Kalman Filter for vehicle tracking process.

3. Faster R-CNN Deep Learning

Using "Faster Region Convolutional Neural Networks" (R-CNN), object detection reduces computational costs and improves accuracy by first using Convolutional Neural Networks (CNN) to classify input image regions and then using R-CNN to identify areas that are likely to be detected. In addition, several region suggestions are produced to improve performance in identifying the intended output. This article detects moving cars on the road using a Faster R-CNN object detector. Reducing time is the primary benefit of requiring training photos, and "Fig. 1" displays the block diagram for the Faster R-CNN model architecture for vehicle identification [19].

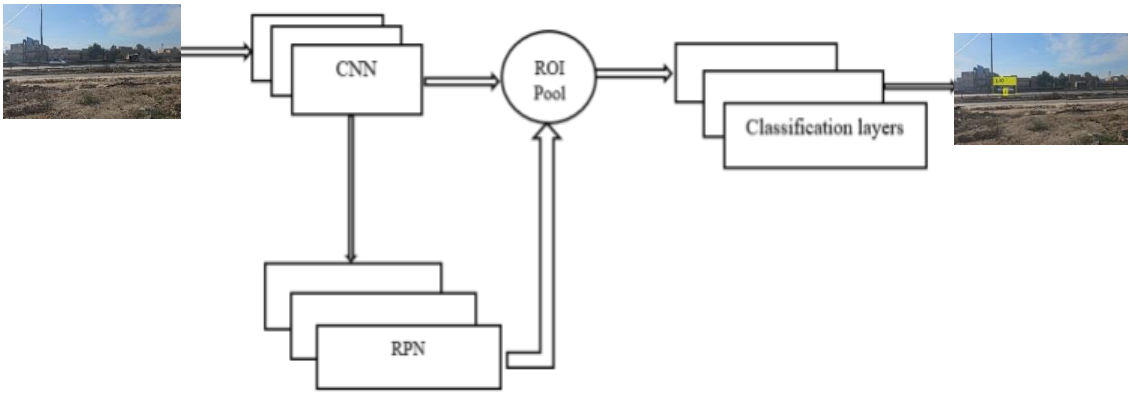

Fig.1. Block Diagram of Faster R-CNN Architecture [19].

An extension of the R-CNN concept, the Faster R-CNN generates a region proposal network rather than learning from any method. Faster R-CNN outperforms R-CNN in terms of runtime performance. Networks for custom region proposals are not supported by this method. The Region Proposal Network (RPN) is used to generate regions and locate objects for object detection. RPN is important because it is a low-cost region proposal and can classify aspects of entire objects or pictures of CNN. The Region of Interest (ROI) pooling, which is frequently used for CNN object recognition, receives input from CNN and RPN. ROI pooling is crucial for reshaping and classifying target areas. The targeted areas are classified in order to produce the final output [19].

The standard Faster R-CNN architecture consists of several key layers and components:

1. **Convolutional Layers**: These layers extract feature maps from the input image. They are typically based on a pre-trained backbone network like VGG16 or ResNet.
2. **Region Proposal Network (RPN)**: This network generates region proposals, which are potential bounding boxes where objects might be located. It consists of:
    - **Anchor Boxes**: Predefined boxes of different scales and aspect ratios.
    - **Convolutional Layer**: Produces feature maps.
    - **Box Regression Layer**: Refines the anchor boxes to better fit the objects.
    - **Objectness Score Layer**: Scores the likelihood of an object being present in the proposed regions.
3. **RoI Pooling Layer**: This layer extracts a fixed-size feature map from each region proposal. It ensures that the features from different-sized proposals are pooled into a uniform size.
4. **Fully Connected Layers**: These layers process the pooled features to classify the objects and refine the bounding boxes further.
5. **Classification Layer**: This layer assigns a class label to each region proposal.
6. **Bounding Box Regression Layer**: This layer refines the coordinates of the bounding boxes to better fit the detected object

And training option:

- **'sgdm'**: Specifies the stochastic gradient descent with momentum optimizer.
- **'MaxEpochs', 60**: Sets the maximum number of epochs to 60.
- **'MiniBatchSize', 32**: Sets the mini-batch size to 32.
- **'InitialLearnRate', 1e-3**: Sets the initial learning rate to 0.001.
- **'CheckpointPath', tempdir**: Specifies the directory where checkpoints are saved.
- **'Plots', 'training-progress'**: Enables the display of training progress plots.
- **'ExecutionEnvironment', 'auto'**: Automatically selects the execution environment (CPU or GPU).

4. Kalman Filter Target Tracking

A linear dynamic system's state may be predicted from observations using the powerful Kalman filter. The current state estimate is calculated recursively, requiring accurate measurement data and state-space estimation just once.

When dealing with Gaussian noise in navigation difficulties, Kalman filters are especially useful. The Kalman filter is an algorithm used in control theory that generates an accurate forecast of the process's state in order to stop noise. This filter is excellent in a number of ways; it enables predictions for the past, present, and future. Even though the precise system model is unclear, this is still possible. Another name for the Kalman filter is the optimum stochastic estimator.

The Kalman filter is a technique that reduces the prediction error covariance while predicting state variables of a distinct linear stochastic dynamical system. A technological method for estimating the function of parameters in time series estimation is the Kalman filter. The KF method's ability to forecast a condition using the least amount of data is one of its advantages. The measuring device's measurement data are the minimal data in question. It is a technique that blends measurements with Kalman filter models. The estimate findings are corrected using measurement data. As a result, the estimate findings consistently become closer to actual circumstances.

In mathematical procedures using Kalman filters, Kalman Gain plays a significant role. The gain is computed using measurement and estimation error data. After that, the immediate estimate is computed. The measured value and the prior anticipated value are utilized in this procedure. The updated estimate's error value is then determined. In the literature, the term "defect definition" is occasionally used to refer to uncertainty [20].

5. Methodology

This work aims to create a system to detect and track moving vehicles to estimate their speed and thus determine high speeds, and this requires several basic requirements, including software, using the MATLAB R2023b programming language, computer specifications: CPU: 13 Gen Intel(R) Core (TM) i5-13400F, Memory: 32GB DDR5 4800 MHz, Graphics Processing Unit: NVIDIA GeForce RTX 3080 12 GB, and in the process of recording data was done using the camera of the Samsung S23 Ultra phone. As shown in Table 1 below:

Table 1. Describe the Tools used in this paper

NO	TOOLS	SPECIFICATIONS
1	Camera	Samsung S23 Ultra phone camera
2	Computer	Cpu :13 Gen Intel(R) Core (TM) i5-13400F
		Memory: 32GB DDR5 4800 MHz
		GPU: NVIDIA GeForce RTX 3080 12 GB
3	Programming language	MATLAB R2023a

The dataset was collected by recording roadside traffic at different times and at different speeds to measure the vehicle's current speed. 20 videos were filmed in Baghdad in the Ghazaliya area using side photography and a database was prepared for this work, 10 of them were cut into frames, and 250 frames were obtained and saved for labeling (Fil_Lbl) to be used in the training process. The rest of the videos were cut into 280 frames and saved in a

test folder (Fil_Tst) to be used in the testing process. The Faster R-CNN model is used for vehicle detection, which is a more accurate and high-speed algorithm model based on a convolutional neural network.

- **Training**: The training algorithm used in this study is as follows:

Algorithm I: Training
INPUT: labeling file (Fil_Lbl), training dataset, training options, and pretrained network (resnet-50)// transfer learning.
OUTPUT: The training model (Mdl).
1 *Perform Faster-RCNN training using inputs to get training model (Mdl).*
2 *Save Mdl to be used later in testing procedure.*
3 ***End***
Where the main training options parameters are:

1. Learning rate (lr=0.001).
2. Epoch (ep=40,50,60,75,90) to get model for each one.
3. Batch size (bs=32).

Algorithm *I* was applied to obtain 5 different training models for each epoch value, (Mdl_40, Mdl_50, Mdl_60, Mdl_75, Mdl_90). Table 2 illustrate the precision training and epoch for each model.

Table 2. shows the precision value of each epoch.

EPOCH	PRECISION
40	98%
50	99%
60	100%
75	99%
90	98%

- **Testing**: The proposed system was tested and applied to the test folder (Fil_Tst) for vehicle detection, The testing algorithm used in this study is as follows:

Algorithm II: Testing
INPUT: Testing file (Fil_Tst) from test dataset, and recognition model Mdl getting from ***Algorithm I.***
OUTPUT: Recognition metrics parameters (TP, TN, FP, and FN).
1 *Recognized vehicle using the model (Mdl).*
2 *Calculate Accuracy criteria.*
3 ***End***

Algorithm *II* was applied 5 times for each model obtained with a change in the epoch value from Algorithm *I*. The accuracy criteria for each model were calculated as shown in the following table 3:

Table 3. shows the accuracy criteria

EPOCH	TP	FP	TN	FN
40	283	3	6	72
50	282	2	6	74
60	308	0	6	50
75	317	3	6	38
90	313	4	6	41

From the results obtained, shown in Table 3, it is clear that the best model is Mdl_75. Therefore, it was adopted in this study to detect vehicles and then track them to estimate their speed.

- **Tracking**: After the vehicle is detected, it is identified by a box and we find the center of this box (Cx, Cy) in the successive frames. Kalman filter is used to perform the tracking process as Kalman filter assigns a single ID to each detected target and then predicts the next location of that target and finally updates the location as shown in the following algorithm:

Algorithm III: Multiple Vehicles Tracking Using Kalman Filter

INPUT: video of vehicle movement on the road, create *vision.KalmanFilter* using *onfigureKalmanFilter*, use prediction and correction methods in sequence to eliminate the noise present in the tracking system, use prediction method to estimate the location of the vehicle when it is occulted by another vehicle.

OUTPUT: For each detected vehicle, its information is stored in an array to allow for accurate tracking over time. This information includes the unique object identifier (*ID*) - coordinates (*Cx,Cy*) - vehicle size - video frame number - time taken to detect **(***ms***).**

initialization of structure for storing tracks *//define all necessary variables.*
1 **initialization of parameters for the Kalman Filters** *// define all necessary parameters.*
// loop runs the video, detects moving vehicles in the video, and tracks them
2 *across video frames.*
while (I <= size) *//read a video frame and detect* vehicles *in it.*
3 ***Detection****//Detection of the vehicle in the frame****.***
4 *// predict new position for all existing tracks.*
5 *// calculate cost for assigning detections to tracks.*
6 *//update assigned tracks.*
7 ***confirmedTracks = updateTracks(tracker, detections, frameCount)*** *//run the tracker on the preprocessed detections.*
8 ***displayTrackingResults(videoObjects, confirmedTracks, frame, mask****)// Display the tracking results on the video.*
9 *//delete tracks that were unassigned for too long time.*
10 ***End***

After tracking the vehicle and determining the center (Cx, Cy) in successive frames, we obtain the vehicle's motion vector on the road as shown in Figure 2. Note that the Y coordinates are fixed because the vehicle's path is almost a straight line, and the X coordinates represent the vehicle's motion vector on the road. Then the distance the car travels between each frame is calculated and divided by the time taken to get the vehicle's speed, but this speed is in pixels and we need it in meters, so we need an additional algorithm to convert the data from pixels to meters, and here we need to calculate the scale factor.

Fig.2. Shows the approximately linear path of the vehicle and specifies the speed in pixel/sec.

- **Convert pixel units to meters:** In this study, a picture of the scene was taken at the size of (1084x1920x3) and two distances were determined for two areas for roads 1 & 2, as in Figure 3. The first distance is Pr(1) = 1 meter and the second is Pr(2) = 4 meters for roads 1 & 2, respectively. To determine these distances in pixels (Px(1) & Px(2)), manual entry is required by double-clicking on the two distances, and thus it is possible to calculate scene length in pixels using the following equation:

$$Px(1) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

$$Px(2) = \sqrt{(x3 - x4)^2 + (y3 - y4)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (7)$$

Here, (x1, y1), (x2, y2), (x3, y3), (x4, y4) represents the coordinates of the clicked points in the image outline, Px(1) & Px(2) determine the distances in pixels. The next step involves calculating the scaling factor (SCF). The scaling factor can be used to convert measurements from pixels to real-world units (meters), and can be calculated using the following equation:

$$SCF(i) = \frac{\mathrm{Pr}(i)(meter)}{\mathrm{Px}(i)(pixel)} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (8)$$

Where i = 1, and 2. Pr(i) represents the distance in real-world units (meters), and Px(i) represents the corresponding length measured in pixels. This scaling factor can then be used to convert the measurements in the image Px(i) from pixels to real-world units (meters, and kilo meters), using the following equations:

$$Pm(i) = SCF(i) * Px(i) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (9)$$

$$\mathrm{Pkm(i)} = \mathrm{SCF(i)} * \mathrm{Px(i)} * \frac{60^2\ \mathrm{hour}}{1000\ \mathrm{km}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (10)$$

Now, to increase the accuracy and speed of detecting moving vehicles on the road, the frame of the video size has been reduced to a third (0.3), so it was necessary to calculate the scale factor after performing the size reduction process. For this, a new algorithm was used to obtain the best-fitting model for scale factors, which allows the scaling factor to be calculated at each image size:

Creating two SCF models Algorithm

INPUT: Picture of the scene (Img), at size (1084x1920x3). Two distances were determined for two areas: Pr(1) = 1 meter for road-1 and Pr(2) = 4 meter for road-2, as in Figure 2.

OUTPUT: Two scale factors models M_SCF1 & M_SCF2 for road-1 and road-2 respectively.

1 **initialization of variables** *//define all necessary variables.*
2 ***while (I <= size)*** *//loop I to visit all size.*
3 ***Show image*** *//get image from source folder and print to screen.*
4 ***Resize image****//resize image to avalue i.*
5 ***Select 4 point (Pr(1), Pr(2))*** *//select two distances points by click mouse.*
6 ***Compute the distance*** *// length of the plane object by using eq(1).*
7 ***Compute the scale factor*** *// calculate the scale factor by using eq(2).*
8 ***End***

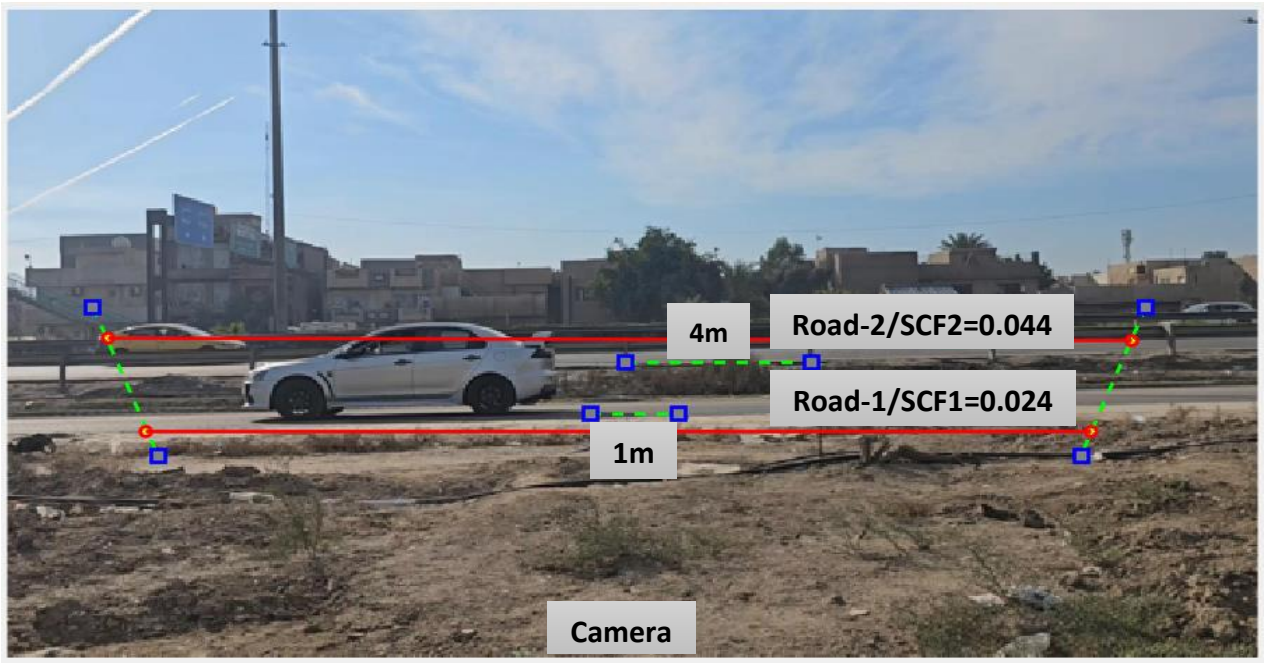

Fig.3. the study scene.

The input image (Img) is resized from i=0.1 to i=2 Step 0.1 and different distances (Pr(1) & pr(2)) were calculated, then for each distance calculate the scale factor (SCF1 & SCF2) using algorithm-1. Furthermore, these results were analyzed, revealing an exponential relationship between the size image and scale factor, see Figure (3). Now by performing curve fitting for these curves get the two models (M_SCF1 & M_SCF2):

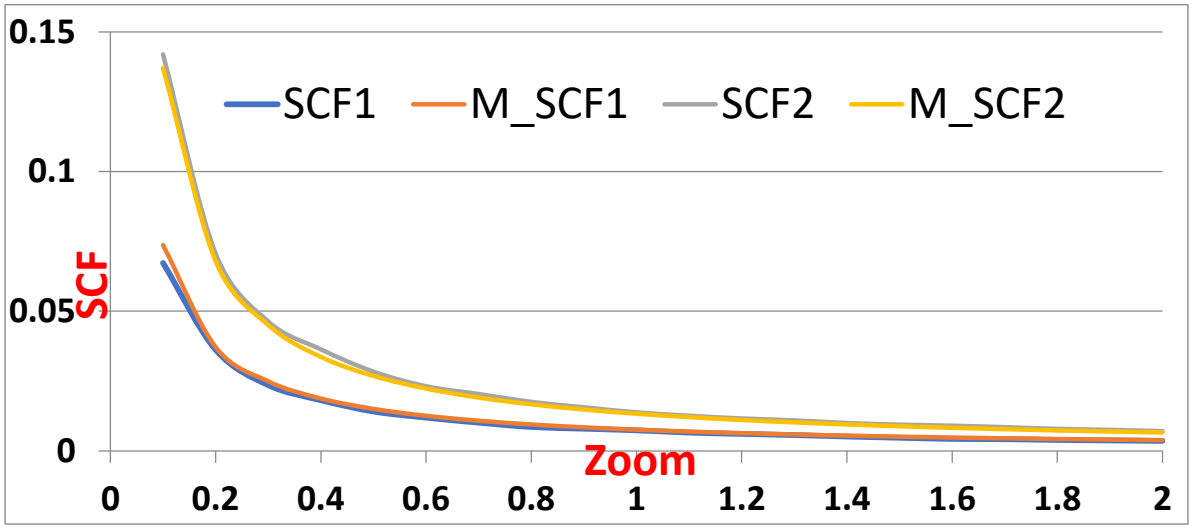

Fig.4. scale factor models

The scale factor is calculated using the two models proposed in this study when the image is

M_SCF1(x) = a*x^b
Coefficients (with 95% confidence bounds):
a = 0.007577 (0.006512, 0.008641)
b = -0.9878 (-1.014, -0.9615)
If x=0.3 then:
SCFm1(0.3) = 0.007577 * 0.3 ^ (-0.9878)
= 0.025

M_SCF2(x) = a*x^b
Coefficients (with 95% confidence bounds):
a = 0.01329 (0.01213, 0.01445)
b = -1.013 (-1.032, -0.9947)

SCFm2(0.3) = 0.01329 * 0.3 ^ (-1.013)
= 0.045

reduced to a third as follows:

6. Results

Several videos were recorded and cut into frames. The frame size was (1084x1920x3). To increase the detection speed, the frame size was reduced to a third, so the frame size became (326x576x3).The study area includes two vehicle roads, on the first road SCF1 was used, and on the second road SCF2, the camera was placed aside and the camera's viewing angle was as shown in figure 2, the vehicle's width (Vw) on the two roads was calculated, as well as the length of the road (Lr) in pixels and meters, and then the number of frames (Nf) and the time (T) of the vehicle's appearance at the scene were estimated when the vehicle's speed (Vs) was 30 km/h, as follows:

Table 4. Theoretical calculations at speed 30 km/h

Road-1 SCF1=0.025	Road-2 SCF2=0.045
Vw = 142.3965 p => Vw = 3.4175 m	Vw = 67.6513 p => Vw = 2.9767 m
Lr1 = 473.0169 p => Lr1 = 11.7757 m	Lr2 = 475.0515 p => Lr2 = 21.3368 m
Let vehicle's speed Vs = 30 km/h, then	
Vs = 30*1000/60/60 = 8.3333 m/sec	
T1 = 11.7757 / 8.3333 = 1.4131 sec	T2 = 21.3368 / 8.3333 = 2.5604 sec
Nf1 = 1.4131 * 60 = 84.7860 frames	Nf2 = 2.5604 * 60 = 153.6240 frames

Table 4 illustrate the calculation of the number of frames and the time taken for the vehicle to

leave the scene when the vehicle's speed is 30 km/h theoretically in both road 1 & 2. It is clear that the number of frames and the time taken in road 1 are less than in road 2, because road 1 is shorter than road 2. In the same way, the above calculations can be repeated for different vehicle speeds and the following values can be obtained:

Table 5. Theoretical calculations at different speeds

Road-1 SCF1=0.025				Road-2 SCF2=0.045			
Vs km/h	Vs m/sec	T1	Nf1	Vs km/h	Vs m/sec	T2	Nf2
30	8.33	1.4131	84.7848	30	8.33	1.4131	153.6252
40	11.11	1.0598	63.5886	40	11.11	1.0598	115.2189
50	13.89	0.8478	50.8709	50	13.89	0.8478	92.1751
60	16.67	0.7065	42.3924	60	16.67	0.7065	76.8126
70	19.44	0.6056	36.3363	70	19.44	0.6056	65.8394
80	22.22	0.5299	31.7943	80	22.22	0.5299	57.6095
90	25.00	0.4710	28.2616	90	25.00	0.4710	51.2084
100	27.78	0.4239	25.4354	100	27.78	0.4239	46.0876

Below are some experimental results obtained from the implementation of the proposed system using Mdl_75, SCF1 and SCF2 at different speeds on roads 1 and 2.

Table 6. Experimental results at different speeds

Road-1, SCF1=0.024				Road-2, SCF2=0.044			
Vs km/h	Vs m/sec	T1	Nf1	Vs km/h	Vs m/sec	T2	Nf2
27.6	7.67	0.3	85	40	11.1111	1.8676	112.0537
29.23	8.12	0.3	85	50	13.8889	1.4940	89.6430
29.23	8.12	0.52	85	55	15.2778	1.3582	81.4936
31.07	8.63	0.28	85	60	16.6667	1.2450	74.7025
39.07	10.85	0.3	65	70	19.4444	1.0672	64.0307
44.42	12.34	0.28	55	75	20.8333	0.9960	59.7620
57.89	16.08	0.27	40	80	22.2222	0.9338	56.0269
58.53	16.26	0.26	45	90	25.0000	0.8300	49.8016

By comparing Tables 5 and 6, notice a great similarity in the values of the number of frames (Nf1 & Nf2) at the different speeds in km/h on both roads 1 and 2, which indicates the efficiency and accuracy of the proposed system. After applying the proposed system to our recorded dataset in Baghdad Iraq some results of detecting and tracking vehicles on roads 1 and 2 are shown below:

a. Single vehicle:

Single vehicle on road 1 detected and tracked using Faster RCNN and Kalman Filter

tracking.

Fig.5. single vehicle on road 1

Single vehicle on road 2 detected and tracked using Faster RCNN and Kalman Filter

tracking.

Fig.6. single vehicle on road 2

b. Multi vehicle:

Multi vehicle on road 1 detected and tracked using Faster RCNN and Kalman Filter tracking.

Fig.7 Multi vehicle on road 1 and road 2

Fig.8 Multi vehicle on road 1 and road 2

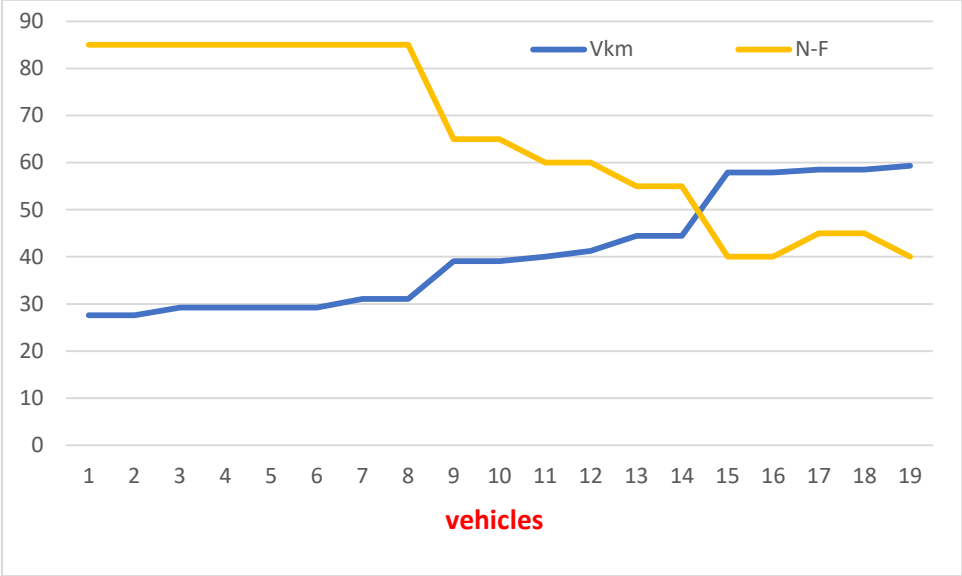

Fig.9.The relationship between NF and the vehicle speed on Road 1

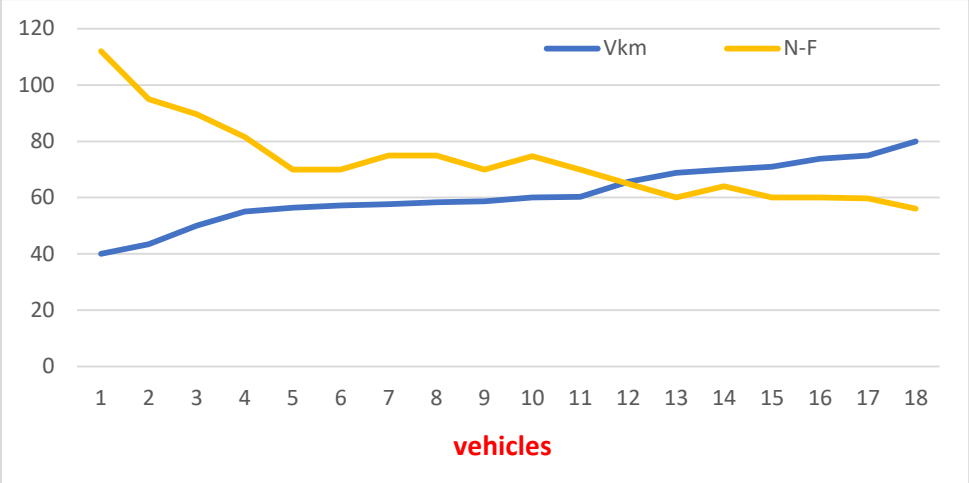

Fig.10. The relationship between NF and the vehicle speed on Road 2

Figures 9 and 10 show the relationship between the number of frames (Nf) and the vehicle speed (km/h) for the remaining vehicles recorded in this study on roads1 and 2, respectively. Note that there is an inverse relationship; as the vehicle speed increases, the number of frames the vehicle must travel to exit the scene decreases.

7. Conclusion

Maintaining active transportation requires precise and consistent traffic flow monitoring. An alternative with 100% accuracy and real-time capabilities has been shown by study, replacing earlier solutions like the pneumatic tube that were used to manually count the traffic flow on certain roadways. Convolutional neural network frameworks and algorithms, such Faster RCNN, offer a very quick and precise technique to count vehicles in a variety of situations, including intersections, basic multi-lane roads, and counting specific lanes inside large junctions. The idea that a basic webcam mounted on a light pole may be able to recognize license plates, track vehicle speeds, count traffic in both directions, and do a lot more than what is feasible with such a basic technology is incredibly attractive.

References

[1] A. Reyes-muñoz and J. Guerrero-ibáñez, "Vulnerable Road Users and Connected Autonomous Vehicles Interaction: A Survey," Jun. 01, 2022, *MDPI*. doi: 10.3390/s22124614.

[2] A. Kumar, P. Khorramshahi, W.-A. Lin, P. Dhar, J.-C. Chen, and R. Chellappa, "A Semi-Automatic 2D solution for Vehicle Speed Estimation from Monocular Videos."

[3] D. Sudha and J. Priyadarshini, "An intelligent multiple vehicle detection and tracking using modified vibe algorithm and deep learning algorithm," *Soft comput*, vol. 24, no. 22, pp. 17417–17429, Nov. 2020, doi: 10.1007/s00500-020-05042-z.

[4] Y. Chen and W. Hu, "A video-based method with strong-robustness for vehicle detection and classification based on static appearance features and motion features," *IEEE Access*, vol. 9, pp. 13083–13098, 2021, doi: 10.1109/ACCESS.2021.3051659.

[5] Q. Tan, J. Ling, J. Hu, X. Qin, and J. Hu, "Vehicle Detection in High Resolution Satellite Remote Sensing Images Based on Deep Learning," *IEEE Access*, vol. 8, pp. 153394–153402, 2020, doi: 10.1109/ACCESS.2020.3017894.

[6] Zheng Tang, Gaoang Wang, Hao Xiao, Aotian Zheng, Jenq-Neng Hwang "Single-camera and inter-camera vehicle tracking and 3D speed estimation based on fusion of visual and semantic features".

[7] K. J. Kim, P. K. Kim, Y. S. Chung, and D. H. Choi, "Multi-Scale Detector for Accurate Vehicle Detection in Traffic Surveillance Data," *IEEE Access*, vol. 7, pp. 78311–78319, 2019, doi: 10.1109/ACCESS.2019.2922479.

[8] J. Zhang, W. Xiao, B. Coifman, and J. P. Mills, "Vehicle Tracking and Speed Estimation from Roadside Lidar," *IEEE J Sel Top Appl Earth Obs Remote Sens*, vol. 13, pp. 5597–5608, 2020, doi: 10.1109/JSTARS.2020.3024921.

[9] M. Famouri, Z. Azimifar, and A. Wong, "A Novel Motion Plane-Based Approach to Vehicle Speed Estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 4, pp. 1237–1246, Apr. 2019, doi: 10.1109/TITS.2018.2847224.

[10] S. Trubia, A. Severino, S. Curto, F. Arena, and G. Pau, "Smart roads: An overview of what future mobility will look like," *Infrastructures (Basel)*, vol. 5, no. 12, pp. 1–12, Dec. 2020, doi: 10.3390/infrastructures5120107.

[11] M. Elassy, M. Al-Hattab, M. Takruri, and S. Badawi, "Intelligent transportation systems for sustainable smart cities," Jun. 01, 2024, *Elsevier Ltd*. doi: 10.1016/j.treng.2024.100252.

[12] J. Hemanth *et al.*, "Object Detection in Deep Surveillance," 2021, doi: 10.21203/rs.3.rs-901583/v1.

[13] R. A. Harianto, Y. M. Pranoto, and T. P. Gunawan, "Data Augmentation and Faster RCNN Improve Vehicle Detection and Recognition," in *3rd 2021 East Indonesia Conference on Computer and Information Technology, EIConCIT 2021*, Institute of Electrical and Electronics Engineers Inc., Apr. 2021, pp. 128–133. doi: 10.1109/EIConCIT50028.2021.9431863.

[14] R. F. Mansour, J. Escorcia-Gutierrez, M. Gamarra, J. A. Villanueva, and N. Leal, "Intelligent video anomaly detection and classification using faster RCNN with deep reinforcement learning model," *Image Vis Comput*, vol. 112, Aug. 2021, doi: 10.1016/j.imavis.2021.104229.

[15] R. M. Alamgir *et al.*, "Performance Analysis of YOLO-based Architectures for Vehicle Detection from Traffic Images in Bangladesh," Dec. 2022, doi: 10.1109/ICCIT57492.2022.10055758.

[16] P. Premaratne, I. Jawad Kadhim, R. Blacklidge, and M. Lee, "Comprehensive review on vehicle Detection, classification and counting on highways," *Neurocomputing*, vol. 556, Nov. 2023, doi: 10.1016/j.neucom.2023.126627.

[17] H.J. Kadhim , A.H. Abbas "A review: The Self-Driving Car's Requirements and The Challenges it Faces" MJPAS Vol. 2, No. 1 (2024) 134-150.

[18] Zahraa Haimeed Rasool , Maha Adham Abdel Amir "Comprehensive Image Classification using Hybrid CNN-LSTM Model with Advanced Feature Extraction on Coco Dataset" MJPAS Vol. 2, No. 2 (2024) 28–47.

[19] Y. Xiao, X. Wang, P. Zhang, F. Meng, and F. Shao, "Object detection based on faster r-cnn algorithm with skip pooling and fusion of contextual information," *Sensors (Switzerland)*, vol. 20, no. 19, pp. 1–20, Oct. 2020, doi: 10.3390/s20195490.

[20] G. Ünal and M. Bilgisi Öz, "Visual Target Detection and Tracking Based on Kalman Filter," 2021. [Online]. Available: https://orcid.org/0000-0001-8285-0954