EasyChair Preprint

№ 10567

Truck Speed Detection Through Video Streams

Zuo Huang, Richard O. Sinnott and Kris Ehinger

September 4, 2023

Truck Speed Detection through Video Streams

Zuo Huang, Richard O. Sinnott, Krista A. Ehinger
School of Computing and Information Systems
The University of Melbourne
Parkville, Australia
zuhuang@student.unimelb.edu.au, {rsinnott,kris.ehinger}@unimelb.edu.au

***Abstract*—Accurately assessing the speed of vehicles is important for traffic management systems. This is especially the case for heavy goods vehicles such as lorries/trucks, since they cannot easily stop at short notice. Previous work has shown that deep learning can be used for identifying and distinguishing trucks on the road from other vehicles, e.g., [1], however accurately estimating their speed from roadside cameras remains a challenge. One solution we employ is using video data from the roadside cameras, then extracting the speeds of vehicles in the video from the Infra-Red Traffic Logger (TIRTL) systems, which are provided by the Department of Transport, Victoria. The TIRTL system is very accurate but expensive and only deployed at a few key locations around Melbourne. A solution that works at the edge and uses lightweight Internet-of-Things devices to produce accurate speed data is thus highly desirable. In this paper, we propose a Convolutional Neural Network (CNN) model using a light-weight Siamese backbone and associated feature correlations to track and detect the speed of trucks. We build a dataset that contains images with speed and bounding-box annotations to train the proposed model. To enable the model to maintain a high degree of accuracy with different camera setups, we train and test the proposed model using image augmentation. The results show our model has an average speed estimation error of 4.92% and an average Intersection over Union (IoU) of 75.8% whilst incorporating different intrinsic and extrinsic parameters based on image augmentation. Such a capability has the potential to change the way services are deployed across the road network to record vehicle types and speeds.**

***Index Terms*—Vehicle speed detection, Deep learning, Convolutional neural network.**

I. Introduction

Vehicle speed detection plays important roles in traffic management, law enforcement, safety and the environment. There are two main types of speed detection methods: *active* and *passive*. Active methods require special hardware such as induction-loops [2], radar [3], lasers [4] or infra-red lighting (TIRTL) [5] to record the timestamp, distance and hence speed of vehicles. Such devices make active speed detection methods possible, however, they are expensive to purchase, deploy and maintain, with devices typically costing hundreds of thousands of dollars (in the case of TIRTL). Internet of Things (IoT) and smaller scale Internet-enabled cameras offer a much more flexible and scalable solution, however establishing the speed of vehicles from such cameras has previously been difficult without a major effort in camera calibration, e.g., recording the distance and angle to the road. Tackling this issue is the goal of this paper. Namely, we want to estimate the speed of specific vehicles (trucks and trailers) from video streams using real world traffic flows from the motorways of Melbourne on cameras that have no calibration.

In contrast to active methods, most passive speed detection methods, e.g., based on video from roadside cameras, use pixel displacements between adjacent frames [6]–[11]. Such methods normally require knowledge of the accurate correlation between the sizes of objects in an image and their real-world counterparts, e.g., some object of known size is used to estimate the distances moved based on pixel displacement between frames. However, the correlation changes if the location and setup of the camera change. Therefore, a speed detection method for monocular camera surveillance systems requires camera calibration to produce correct mapping if intrinsic and extrinsic parameters of the camera change, e.g., the perspective, scale, rotation, and location.

Automatic camera calibration methods for speed detection such as [12]–[14], use the vanishing points and scale of the image to produce the intrinsic and extrinsic parameters of the cameras. However, it is impossible to estimate the scale of an image if the size of objects is unknown [14]. To tackle this problem, one method is to include (calibrate) the related dimension information of vehicles or lanes manually [15]. This may or may not be possible with IoT cameras or it may be the case that this requires considerable manual intervention. Another method [12] is to employ detector to detect the type and bounding-boxes of vehicles, then use the normal/standard dimension information of vehicle types, e.g., specific commonly occurring vehicles are known to be of a given size. This is unlikely to work in all situations however, e.g., where no instances of those vehicles occur in the video. Generally, camera calibration methods need external information to estimate the scales of images, and this is problematic and leads to inflexible solutions that will only work in one, manually established and calibrated situation.

In this work, we present a deep learning model capable of detecting the speed of trucks without the need for camera calibrations. Relatively little effort has been made to estimate speed using deep learning applied to video streams. It is difficult to integrate current camera calibration methods into deep learning speed detection models, e.g., based on Convolutional Neural Networks (CNN), since it is not possible to know which parameters in the CNN backbone represent the intrinsic and extrinsic parameters of a given camera. Some works such as [7], [16] employ deep learning models to detect and track vehicles in videos to get the location information

such as bounding-boxes, and then estimate the speed using pixel displacements methods, however as noted, this requires models to have knowledge of accurate correlation between the sizes of objects in an image and their real-world counterparts. To tackle this problem, [16] measures the distance between vehicles based on real-world coordinates, while [7] assumes the maximum speed information of the roads is known.

Assumptions on camera calibration makes deep learning models difficult to apply to estimate the speed of vehicles directly. An alternative approach is to train CNN models with data sets that have images or videos taken from different camera setups. Barros et al. [17] propose a deep learning model trained using a synthetic data set based on the CARLA simulator. They simulated different light and weather conditions for different vehicles in the data set with the same setup parameters for the cameras in different scenarios. This included fixed overhead cameras positioned to view the rear of vehicles. The camera setup of the validation data set [18] was also fixed. Although the authors claimed their model could avoid the need for camera calibration, their work was somewhat limited, e.g., it did not provide experimental results dealing with changes of rotation, perspective, shift and scale - all of which are necessary for real world camera setup scenarios to measure the speed of vehicles, e.g., it is typically not the case that cameras can be placed above the road to record vehicles.

The speed of heavy vehicles is an important factor that can be used to indicate congestion, traffic jams, speeding violations as well as air pollution, e.g., slow moving vehicles in traffic jams will give rise to more localised pollution. In this work, we propose an end-to-end deep learning speed detection model that can not only track and estimate the speed of a given heavy vehicle, but also can adapt to the change of setup parameters of cameras automatically without the requirement for camera calibration. To achieve this, we build a data set that has highly accurate bounding-boxes and associated speed annotations for image pairs extracted from road cameras. To produce the bounding-boxes and identify the specific types of trucks, we employ a state-of-the-art deep learning heavy truck classifier [1] trained with a data set comprising images captured from cameras on multiple road conjunctions in Melbourne. The mean Average Precision (mAP) of the classifier reached 79.09% for the different truck types across all sites. For the speed annotations, we employ the data captured from the TIRTL detection system, which has a speed detection error less than 1% [5].

To enable the proposed model to estimate speed correctly in different camera setups, we employ augmentation technologies to simulate different camera settings with a focus on varying perspective, shift, scale, rotation, light, and diverse weather conditions. We hypothesize that a CNN model can learn automatically the correct mapping between the sizes of objects in an image and their real-world counterparts by training it on a data set containing images with varying intrinsic and extrinsic parameters, since the mapping is highly correlated with factors such as truck type, bounding box, travel distance, and width of the lane, among others. Based on this hypothesis, we designed an associated model and ran a range of evaluation experiments.

Specifically, the proposed deep learning model includes two parts: tracking vehicles and speed estimation. Inspired by SiamBan [19] which utilises a Siamese architecture [20] as the backbone to track objects, we employ a three-branch Siamese architecture where all of the branches share the same parameters as the backbone to produce the feature maps. For tracking, the model uses a depth-wise separable correlation block [19], [21] to produce cross-correlation features between trucks in video frames for estimating the bounding boxes. For speed estimation, inspired by FlowNet [22], we use spatial correlations [22] between frames as features, from which the model estimates the speed of truck by using a full connection block.

This paper makes the following contributions:

- we propose an end-to-end model that can not only track target trucks from diverse traffic flows, but also estimate the speed of these trucks in videos based on deep learning methods;
- we establish a data set with highly accurate speeds and bounding-box annotations that is suitable for training and evaluation of the deep learning model;
- the proposed model achieves an average error of 4.1% for speed estimation and average Intersection over Union (IoU) of 79% for bounding-box estimation;
- the proposed model achieves an average speed estimation error of 4.9% when the images are augmented to reflect different weather, light conditions, perspective, shift, scale and rotations.

Our work has several known limitations. First, the proposed model focuses on tracking and detecting the speed of a single heavy vehicle. As a result, the model requires an additional truck detector to detect a given truck and obtain its bounding box in order to initiate inference from the real-world video. Second, the data augmentation methods we used are not capable of representing all intrinsic and extrinsic parameters of images in real-world scenarios, e.g., zooming in/out. Third, the proposed training and test data sets are generated from at present a single site, and hence the augmentation methods may not be able to accurately detect the speed of vehicles from other sites with complete accuracy. This is primarily because we require gold standard speed recordings, which are only available through systems such as TIRTL. As noted these are expensive and immobile and will never be deployed at scale across the complete road network. Despite these limitations, the work illustrates a viable model that could form the basis for widespread service delivery model for vehicle classification and speed estimation in IoT environments. It is also noted that at present the Department of Transport in Victoria is not allowed to store video streams in a centralised repository due to privacy legislation in Australia. Hence vehicle classification and reporting on public street networks demands IoT-based services/solutions that can process the data locally.

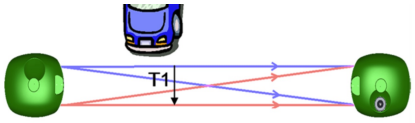

Fig. 1. Overview of Speed Detection based on the TIRTL System [5]

II. Related Works

We begin this section with a discussion of the TIRTL technology, which is an active method used to collect vehicle speed data. We then review passive methods for detecting vehicle speed in videos. In terms of passive speed detection methods, there are two types: non-deep learning methods and deep learning methods. We also discuss methods for camera calibration, as most passive speed detection methods require camera calibration to be completely accurate.

A. TIRTL

In this work we utilise ground truth speed and vehicle classification data from the TIRTL system. Fig. 1 illustrates the basic TIRTL set up. Specifically, the TIRTL system transmits two infra-red light beams passing slightly above the road surface. TIRTL then uses the times that the first and second parallel beams are blocked and unblocked by the tyres of a given vehicle. The timing difference between the beams is then used to calculate the speed of vehicles based on the distance between the wheels/axels and the time that the beams were broken. A TIRTL system has an accuracy of less than 1% error at up to 250km/h. The Australian National Measurement Institute and The National Association of Testing Authorities (NATA) have accepted TIRTL as the basis for speed detection cameras throughout Australia [5]. However as noted, the TIRTL cameras and the software they use are very expensive and only located in a few locations around Melbourne.

B. Non-Deep Learning Passive Methods

Passive speed detection methods first track objects and then estimate their speed using pixel displacement between adjacent frames in a given video stream [6], [8]–[10].

CVS [10] extracts the foreground information (vehicles) based on the difference between frames and uses edge detection to produce a bounding box around the vehicle and its associated centroid. Using the Euclidean distance of the centroid of the vehicle in adjacent frames, the model estimates the vehicle's speed. Through this approach, CVS achieves an average error of 7km/h.

Some methods utilise segmentation. As one such example, the KNN method [8] first uses K-Nearest Neighbours (KNN) to extract segmentation information from the foreground of images. It then produces a bounding box from the contours of the vehicle segmentation before using centroid pixel distances of the bounding-boxes between two consecutive frames to identify moving vehicles. To get the correct sizes of objects in the image, knowledge of the standardised sizes of road traffic signs and lines (in China) are used. Finally, the model estimates the speeds of vehicles based on pixel displacement of the tracked vehicle. Overall it achieves a relative error of 5%. Whilst the results are positive, they are based on assumptions that mean that the model needs adaptation to other locations (countries), e.g., for different sized signs and lines.

The intrusion-lines method [6] estimates the speed of vehicles based on intrusion lines that are drawn on the ground plane of the image in parallel (not in the real world). The model allocates intrusion lines for images based on the distance in the real-world. The speed can then be calculated based on the number of intrusion lines and the frame rate of the video. The model achieves an average error of 2.17%. However, the accuracy of this method depends on the accuracy of distances between the intrusion lines, which change when the setup of the camera changes.

C. Deep Learning Methods and Data Sets

There are two key data sets that have speed annotations [12], [18]. Luvizon data set [18] has video based data samples with associated speed annotations. These speeds are produced by inductive loop detectors. The videos were captured by a fixed overhead camera. In total 7,766 vehicles with speeds and associated plate numbers were annotated. Sochor data set [12] is for camera calibration, in which three over-head cameras were used to capture 20,865 vehicles with the speed annotations obtained by a pair of LiDAR devices on the side of road. However, these two data sets do not provide bounding-box annotations of the objects moving in the image.

We consider speed detection models that employ deep learning methods for detecting, tracking or computing the speed of vehicles. The detect-then-track method [7] employs a deep learning detector [23] to detect vehicles and produce their associated bounding-boxes. It then employs optical flows as enhancements to track vehicles across adjacent frames. For speed calculation, the model assumes that the maximum speed limit is known and at least one vehicle drives at that speed. Under this assumption, it uses the proportion between the movement of the target vehicle and the maximum local movement to estimate the speed based on the assumed maximum speed. Although this method can avoid camera calibration, the assumption of the maximum speed limit is unlikely to be correct in real-world scenarios.

To avoid camera calibration challenges, Bell et al. [16] employ real-world coordinates to obtain the distance between trucks across frames. The model first detects the vehicle by using a fine-tuned YOLOv2 detector [24], before tracking the vehicle by using the Simple Online and Realtime Tracking (SORT) method [25]. The final speed estimate achieves a mean absolute percentage error of 20.92%.

Deep learning speed estimation models require data sets with annotations for the ground-truth speeds and bounding boxes of the vehicles. To tackle this problem, Barros et al. [17] use a synthetic data set based on the CARLA simulator [26] to train their deep learning model pipeline. They first use a Faster R-CNN model to detect the vehicle, then employ the DeepSORT tracker to track the vehicle. They then use

FlowNet2 [27] to extract the dense optical flow between adjacent frames, before finally feeding the optical flow to a VGG16 CNN to estimate the speed. The benefit of this synthetic data set is that the researchers can build a data set for multiple scenarios, including different light, weather and camera setups. The model was validated using Luzivon's data set [18] and achieved an error range of [-3, + 2] km/h for 85.4% of data samples.

D. Camera Calibration Methods

Many camera calibration methods use vanishing points (VPs) and scale to estimate the intrinsic and extrinsic parameters of given cameras. [28], [29] use a projection matrix $\mathrm{P} = \mathrm{K}[\mathrm{RT}]$ for calibration between two cameras, where T is based on the scale, and R and K are based on the focal length of the camera and other parameters.

There are many methods that have been applied to tackle VPs and scale. For example, Dubska et al. [14] employ Hough transforms to detect three VPs used to capture the motion of a vehicle. The first VP is the direction of the vehicle; the second VP is perpendicular to the first VP (in the ground plane), whilst the third VP is perpendicular to the ground plane. The model detects vehicles to get their bounding-box and type, before using dimension information of the specific types of vehicles to calculate the scale. Other methods, such as [30]–[33] employ road lines or lanes as vanishing points and input the width of lanes or sizes of other objects manually to get the scale of the particular content of images.

III. Proposed Model

The proposed speed detection model includes four parts: a Siamese CNN backbone; two five-layer Depth-wise Separable Correlation Blocks (DSCB) inspired by [19], [21], a Spatial Correlation Block (SCB) inspired by [22], [27] and a Speed Full Connection Block (SFCB) as shown in Fig. 2. The model requires three input images including the first search frame, the target frame and the second search frame. The search frames are the input images. The target is a given truck who's speed is to be estimated. The target frame is a cropped patch of the first search frame based on the bounding box of the target truck. The three images are fed into the triple-Siamese backbone to produce three five-layer feature maps. Each layer of the feature maps is fed into a DSCB to produce a classification map and a regression map. The outputs are then fused (averaged) as shown in Figure 2. The regression maps are then used to produce bounding-boxes of the truck in the search frames. For estimation of the speed, classification map 1 and classification map 2 are fed into the SCB and SFCB. The outputs of the model include two classification maps, two regression maps, and an estimated speed of the target truck. The loss function uses all five outputs to produce the losses for the optimizer during model training.

The classification map itself has two channels: one channel indicates if a pixel belongs to the foreground, whilst the other indicates if a pixel belongs to the background. The regression map has four channels, where each channel has information regarding the distance from a pixel to the coordinates of the predicted bounding-box.

A. Siamese CNN Backbone

The Siamese Backbone includes three identical sub-networks comprising three EfficientNet-Lite4s [34] that share the same weights. From three input images the backbone produces three feature maps. These are used to extract the last five layers of the feature maps as inputs to the DSCBs.

B. Depth-wise Separable Correlation Block

The model has 2x5 depth-wise separable correlation blocks (DSCB) as shown in Figure 2. A DSCB takes a search feature map and a target feature map as inputs and uses them to produce a classification map and a regression map. To achieve this, the DSCB first adjusts the width and height of the input tensor using bilinear interpolation to make sure it has the correct sizes, namely: 44x44 for the search feature map and 7x7 for the target feature map. Since we use EfficientNet as the backbone, the size (width and length) of every layer of the feature map is different. The convolutional layers decrease the number of channels of the tensor by half to reduce the computational cost. The DSCB then feeds the tensors to two depth-wise correlation layers [21] to produce the regression map and classification map. These go through a convolutional layer with 1x1 filters to make sure the output tensor has the correct number of channels: four for the regression map and two for the classification map.

The depth-wise correlation layer is a convolutional operation that uses the search frame feature map as input and target feature map as filter (kernel), such that the output features have a strong correlation to the target image (truck) [21]. Although the depth-wise correction layer is not trainable, the channel adjustment layers are trainable.

C. Spatial Correlation Block

FlowNet [22] and FlowNet2 [27] are deep learning models that produce optical flows from input videos. We employ a Spatial Correlation layer from the FlowNetCorr model [22]. Unlike DSCB that produces correlations between two full feature maps (the target frame and search frame) that are used for tracking, the Spatial Correlation Block produces patch correlations. The spatial correlation between two patches of the feature maps (f_1 and f_2) is defined [22] as:

$$c(x_1, x_2) = \sum_{o \in [-k,k] \times [-k,k]} \langle f_1(x_1 + o), f_2(x_2 + o) \rangle \qquad (1)$$

where x_1 and x_2 are the center points of two patches and $(2k+1) \times (2k+1)$ is the size of the patch. FlowNet uses the spatial correlation $c(x_1, x_2)$ to produce optical flows, while we use it to estimate the speed of vehicles. The value of k decides the number of channels of the output tensor. In this work, we set k to 5.

The spatial correlation layer in FlowNet produces correlations for all objects between two adjacent frames. However, given we want to estimate the speed of specific vehicles

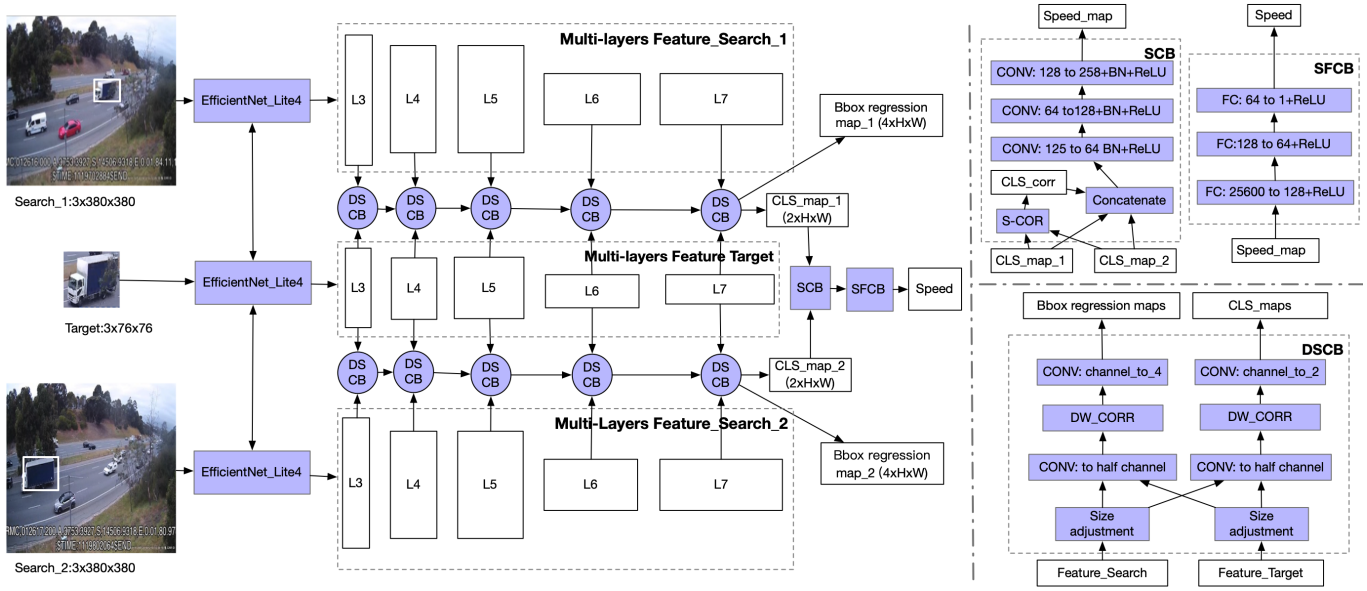

Fig. 2. Proposed model based on the Siamese architecture where all branches of the backbone (EfficientNet-lite4 [34]) share the same parameters (weights). The Right-Hand-Side (RHS) shows the details of the Spatial Correlation Block (SCB), the Speed Full Connection Block (SFCB) and the Depth-wise Separable Correlation Blocks (DSCB) [21]. DW_CORR is the Depth-wise Correlation Layer, S-COR is the Spatial Correlation Operation (Equation 1), CLS_corr is the output of S-COR, CONV is the convolutional layer, Bbox is the bounding-box, CLS_map is the classification map, BN is the batch normalization, FC is the full connection layer, and ReLU is the Rectified Linear Unit activation.

(trucks), we need the correlation between trucks in two frames to estimate the speed. This requires the model to extract the foreground from the feature map to reduce the noise caused by the background before the correlation operation occurs. In this work, classification maps fulfill this requirement since they have the correlations between the frame and the target truck. The SCB first produces the spatial correlation from these two classification maps based on Equation 1. These then concatenate the output correlation map and all classification maps together, before feeding the concatenated feature map into a convolutional block to produce the speed feature map, as shown in the right-hand side of Fig. 2.

D. Full Connection Speed Block

We use a Full Connection Speed Block (FSCB) to estimate the speed of the target as shown on the right-bottom of Fig. 2. The input of the FCSB is the output of the Spatial Correlation Block. We use Rectified Linear Units (ReLU) as activation functions in every layer of the FCSB to increase the non-linearity.

E. Loss Functions and Temporary Stop Mechanisms (TSM)

We optimize the model using three losses: the classification loss, the bounding-box loss and the speed estimation loss. We employ a cross-entropy loss function to produce the classification loss between the classification map and the ground-truth classification map. We use a Huber loss function to produce the bounding-box loss between the predicted regression map and the ground-truth regression map. We use a smooth L1 loss function to produce the speed estimation loss with inputs including the predicted speed and the ground-truth speed. The loss function itself is given as:

$$L = \alpha L_{cls} + \beta L_{reg} + (1 - \alpha - \beta) L_{sp} \tag{2}$$

where L_{cls}, L_{reg} and L_{sp} are the classification loss, the bounding-box loss and the speed estimation loss respectively. We set $\alpha = 0.3$ and $\beta = 0.3$ as the weights of the losses.

Since back-propagation is guided by three types of loss in training, it is possible that the model could focus on one loss but pay less attention to the other loss types, e.g., the model may do a better job in predicting bounding boxes than predicting speed. To tackle this problem, we introduce a temporary stop mechanism in the training process so that if the average loss of the bounding-box is less than four in a given batch (with a size of 12), we set β in Equation 2 to 0. As a result, the back-propagation of this batch will ignore the bounding box loss, which ensures that the model places more attention on the object speed in the training process.

IV. Experiments and Results

A. Proposed Data Set

In this work, we utilise a data set comprising 2,466 data samples captured from a stretch of major road network (motorway) near Melbourne. Each data sample includes an image pair with associated annotations for given truck types in the image (see Fig. 5). An image pair includes two search frames extracted from the video captured by the road camera as shown on the left-hand-side of Fig. 2. These two frames have time differences of 1 second between them. The annotations include the bounding-box and speed. The bounding-box annotation is produced by using a heavy truck classifier which has a mAP

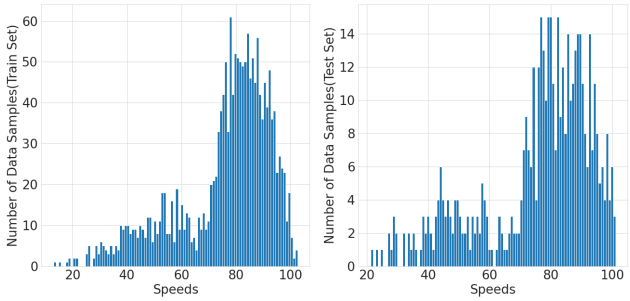

Fig. 3. Histograms of Data Samples based on Vehicle Speed in KM/H. The left-hand side is for the training set and right-hand side is for the test set.

of 79.09% [1]. The ground truth speed annotation is based on the TIRTL laser devices deployed across the motorway.

To build the data set, we first applied the truck classifier [1] on a video stream captured on one specific location in Melbourne to produce the types (numbers of wheels/axels) and bounding-boxes of heavy trucks. We then extract two frames with 1 second time differences and include the same truck from the video. We extract the speed of the trucks from the recordings of the TIRTL system by matching their timestamp and associated vehicle type, e.g., 6-axel truck.

We split the data set into three subsets: a training set with 1,730 data samples; a validation set with 248 data samples and a test set with 488 data samples. See Fig. 3 for the distribution of the data samples, where the distribution was based on the speed of the vehicles in the training and test data sets. The data set includes highly accurate speed and bounding-box annotations, however there are some drawbacks. One drawback of the data set is that the frames are extracted from video captured from a fixed camera in one location. As described, our goal is to produce a model that can detect and classify the speed of trucks using arbitrary surveillance cameras in different locations and at different angles. To tackle this, we apply augmentation techniques to the images during the process of training. A second limitation is that the speed annotation of some data samples may not be accurate, since the location of the speed detection devices is approximately 50 meters before the location of the surveillance camera. This can give rise to a time difference of 3-5 seconds. It is possible that the speed of the truck might change during this time. A third drawback of the data set is that we employ a truck detection model [1] to detect the bounding box of trucks. Although the detection model has a high accuracy, we also conduct manual checking of the data samples. However we cannot guarantee the bounding-box always matches the trucks perfectly.

B. Training, Validation and Testing

We train the models using Nvidia GPU and Intel CPU. To enlarge the size of the training data and simulate images from different cameras, we augmented data set based on data augmentations and associated probabilities as shown in Table I.

TABLE I
AUGMENTATION METHODS. PROBABILITY IS THE PROBABILITY OF TRIGGERING A METHOD. SEE [35] FOR DESCRIPTIONS AND PARAMETERS.

Augmentation Methods	Probabilities
Horizontal Flipping	0.5
Random Brightness Contrast	0.5
Motion Blurring	0.2
Random Fog	0.1
Random Rain	0.3
Random Shadow	0.2
Random Sun Flare	0.2
Perspective	0.7
Shift Scale Rotate	0.5

During training, we use Adam as the optimizer with a learning rate of $2e-5$ and a weight-decay rate of $1e-4$.

For validation, we use a validation data set augmented using the same methods and the same probabilities as the training data set. Ultimately, we select the model which has the lowest validation loss of speed estimation as the optimal model.

In the test stage, we evaluate the best model using both the test data set and the augmented test data set. Since the parameters of the augmentation methods are produced randomly in default ranges defined by [35], the test results (Average Error of Speed as defined by Equation 3) of the augmented test data set may change within a range of $(-0.003, +0.003)$ in every experiment. In this work, we use the mean of the results from multiple experiments as the final result.

C. Evaluation Metrics

For speed evaluation, we use the average error (AE) of speed estimation as the metric. This is based on:

$$AE = \frac{1}{N}\sum_{i=1}^{N}\frac{|S_i^{pred} - S_i^{gt}|}{S_i^{gt}} \tag{3}$$

where N is the size of the test data set, and S_i^{pred} and S_i^{gt} are the predicted speed and ground-truth speed of the i data sample, respectively.

We use the average Intersection over Union (A.IoU) as the key metric to evaluate the accuracy of the predicted bounding-box. The IoU is defined as:

$$IoU = \frac{Area^{overlap}}{Area^{union}} \tag{4}$$

where $Area^{overlap}$ is the area of overlap of the predicted bounding-box and the ground-truth bounding-box, and $Area^{union}$ is the area of the union of both the predicted bounding-box and the ground-truth bounding-box.

D. Results

We evaluate the model performance by using the original test data set and the augmented test data set. Table II shows that our model gets an average speed estimation error of 4.18%, and an average IoU of 0.767 in the original test data set.

TABLE II

COMPARISON WITH OTHER MODELS. THE NUMBERS IN THE ERROR COLUMN WITHOUT KM REPRESENT THE AVERAGE ERROR IN SPEED (BASED ON EQUATION 3). A.IOU MEANS THE AVERAGE IOU FOR OUR MODEL. TEST BY ORIGINAL/AUGMENTED IS THE MODEL TESTED USING THE ORIGINAL/AUGMENTED TEST DATA SET RESPECTIVELY. THE TEST RESULTS OF OTHER MODELS ARE FROM THE LITERATURE THAT HAS BEEN CITED, AND THEY ARE NOT RE-EVALUATED HERE.

	Model	Error	A.IoU
Active	TIRTL [5]	≤ 0.01	-
	Induction-loop [2]	≤ 0.05	-
Passive	CVS [10]	7km/h	-
	Intrusion-lines [6]	0.0217	-
	KNN [8]	≤ 0.05	-
Deep-learning	YOLO+SORT [16]	0.209	-
	Synthetic-data [17]	3km/h in 85.4%	-
Ours	Test by Original	0.0418	0.767
	Test by Augmented	0.0492	0.758

TABLE III

AVERAGE ERROR AND IOU COMPARISON USING THE TEST DATA SET AUGMENTED BY ONLY ONE METHOD, WITH PROBABILITY 1.

Augmentation	Average Error	Average IoU
Horizontal Flipping	0.0443	0.776
Random Brightness Contrast	0.0433	0.770
Motion Blurring	0.0437	0.764
Fog	0.0456	0.759
Rain	0.0460	0.757
Shadow	0.0447	0.771
Sun Flare	0.0529	0.753
Perspective	0.0447	0.751
Shift Scale Rotate	0.0504	0.790

To evaluate how our model maintains accuracy of speed estimation using other images/videos produced by road surveillance cameras with different setup and parameters, we use a test data set augmented by two approaches: the first based on the same methods and probabilities as the training process as shown in Table I, and the second augmented using only one method with probability 1. Fig. 5 shows examples from the augmented data samples. Table II shows that our best model gets an average error of 4.92% in the test data set augmented by the first approach. Table III shows the average errors of the test data set augmented using the second approach.

E. Analysis of Results

The models in Table II and other speed detection methods are evaluated with different test data sets. Such data sets get their speed data using different measurement methods such as GPS devices in vehicles or induction-loop speed detectors. Such evaluation metrics are not unified. As such, comparison of results of speed detection models needs to be treated with care. To evaluate the accuracy of our model, we investigate the average errors based on different speed ranges in the augmented test data set. Table IV shows that our model has average errors of less than 3% when the ground-truth speed is larger than 80km/h, but has higher average errors when the ground-truth speed is smaller than 40km/h. The right-top plot

TABLE IV

AVERAGE ERROR AND AVERAGE IOU COMPARISON BASED ON THE GROUND-TRUTH SPEED IN THE AUGMENTED DATA SET, STD IS THE STANDARD DEVIATION, SP IS SPEED, AND ERR IS ERROR.

GT. Sp	A.Err	STD.Err	MAX.Err	A.IoU	STD.IoU
0-40	0.138	0.101	0.330	0.730	0.107
40-50	0.080	0.066	0.259	0.717	0.138
50-60	0.049	0.061	0.270	0.767	0.101
60-70	0.058	0.038	0.139	0.773	0.098
70-80	0.047	0.048	0.363	0.765	0.105
80-	**0.030**	0.026	0.191	0.772	0.111

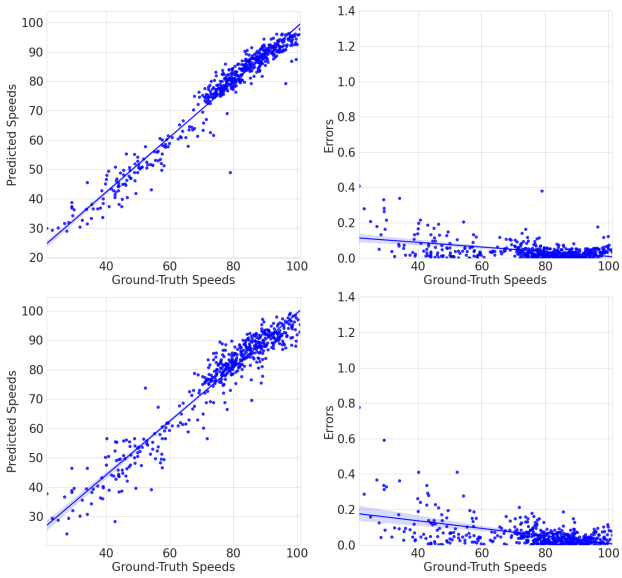

Fig. 4. Results of the Test Data set. The LHS shows the correlation between the predicted and ground-truth speeds. The RHS shows the speed estimation errors of every data sample. The top plots show the results of the non-augmented test data set and the bottom plots show the results using the test data set augmented by Shift Scale Rotate with probability 1.

in Fig. 4 shows the same pattern based on the ground-truth speed of every data set sample.

We also investigate the correlation between the predicted speed and the (TIRTL-based) ground-truth speed of every data sample in the test data set. The left plots in Fig. 4 show a strong linearity between the predicted speeds and the gold standard (ground-truth TIRTL-based) speeds.

F. Correlation between Bounding-box and Speed

It is reasonable to assume that a deep learning model could use many factors to estimate the speed of a target vehicle such as a truck. These factors might include the number of objects, the size and location of the objects, and the surrounding background information and other image resolution challenges, e.g. rain/fog. We consider the speed of target trucks using the size and location of the targets. To evaluate this, we investigate the correlation between the error of speed detection and the IoU of every data sample in the test data set. Fig. 6 shows a strong correlation between the errors of speed detection and IoUs of the bounding-boxes in the test data sets (original and

Fig. 5. Results using the Augmented Data Set. The top images are the first frame, the bottom images are the second frame; EST indicates the estimated speed and GT indicates the ground-truth speed.

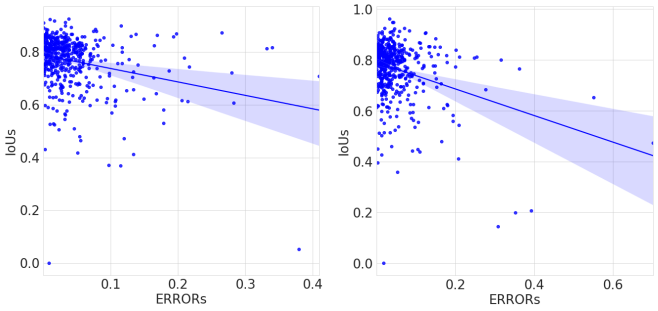

Fig. 6. Correlation between IoUs and Error of Speed. The LHS shows IoU vs Error for every data sample in the non-augmented test data set; the RHS shows IoU vs Error using the data augmented by the methods in TableI

augmented). Most of the dots in the plots fall into areas with low errors of speed and high IoUs. Thus our model can predict speed and bounding-boxes at the same time. It also implies that the background objects only have a weak influence on the speed detection.

G. Inference In Videos

There are two requirements for tackling inference in videos using the proposed model. The first is the frame rate of the video to establish the time difference between frames. This can be obtained from the camera or using software such as OpenCV. The second requirement is establishing the bounding-box of the target truck. In this work, we use the truck detector [1] to get the bounding-box of a truck in the first frame, then feed it along with the image pair after one second into the model to estimate the speed of the vehicle. The benefit of this method is that we do not need to infer the speed in every frame, which makes the model more cost effective.

TABLE V
ABLATION STUDY: COMPARISON BETWEEN MODELS WITH AND WITHOUT SPATIAL CORRELATION BLOCK (SCB).

Model	Average Error	Average IoU
w/o SCB	0.044	0.791
w SCB	**0.041**	0.767
w/o SCB(augmented)	0.067	0.795
w SCB(augmented)	**0.049**	0.758

H. Ablation Study

We also conducted ablation studies to show that the Spatial Correlation Block (SCB) and Temporary Stop Mechanisms (TSM) can increase the accuracy and to check that the model is robust using the augmented test data set. We train a model without Spatial Correlation Blocks then evaluate it using the augmented data sets. Table V shows a significant improvement caused by the SCB. The average error of the model with SCB decreases from 6.7% to 4.9%. However, it causes a decrease in accuracy of the bounding-box estimation.

Table VI shows a significant improvement caused by the Temporary Stop Mechanism (TSM). The best model (with SCB) trained by TSM has lower average speed estimation errors compared to the best model trained without TSM. The decrease in accuracy of the bounding-box estimation is expected since the TSM ensures that the training puts more attention to the speed if the loss of the bounding-box is small enough.

To check the robustness of the model with regards to changes in the camera's parameters, we compare the model

TABLE VI
ABLATION STUDY: COMPARISON BETWEEN MODEL WITH AND WITHOUT TEMPORARY STOP MECHANISM (TSM).

Model w SCB	Average Error	Average IoU
w/o TSM	0.052	0.790
w TSM	**0.041**	0.767
w/o TSM(augmenged)	0.061	0.772
w TSM(augmented)	**0.049**	0.758

TABLE VII
ABLATION STUDY: TEST RESULTS FOR THE AUGMENTED TEST DATA SET WITH DEFAULT SETTING [35], AND THE MODEL TRAINED WITH AUGMENTED AND NON-AUGMENTED DATA SETS

Best model trained by	A.Error	A.IoU
Non-augmented dataset	0.247	0.15
Augmented dataset (default)	**0.049**	0.758

TABLE VIII
ABLATION STUDY: TEST RESULTS FOR THE AUGMENTED TEST DATA SET WITH AGGRESSIVE SETTINGS; COMPARING THE BEST MODEL TRAINED USING THE DATA SET AUGMENTED WITH DEFAULT AND AGGRESSIVE SETTINGS.

Best model trained by	A.Error	A.IoU
Shift:0.0625,Scale:0.1(default)	0.067	0.704
Shift:0.2,Scale:0.5(aggressive)	**0.053**	0.754

trained using the non-augmented data set and the augmented data set (with default parameters) [35]. If the parameter is a float number p in a range of [0,1], the shift method will shift the image by a random factor between 0 and p for both the height and width, while the scale method will zoom in/out the image by a random factor in $[0, p]$ [35]. Table VII shows that our model can adapt to the change of camera setup parameters if it is trained using the augmented data set.

To show that our model can adapt to more aggressive changes in camera setup, we increase the parameter p used for the shift and scale method in the training and test data set as shown in Fig. 7. We ignore the augmentation operation that makes the truck/bounding-box too small (less than 5% of the width or height of the image) or where it places it outside of the image. Table VIII shows that the best model trained by the augmentation methods with default parameters has a worse accuracy based on the test data set augmented with aggressive parameter settings, however, the best model trained using the data set augmented with aggressive parameters achieves a better accuracy.

V. CONCLUSIONS

This paper has made the following contributions:

- we propose a lightweight Siamese CNN backbone to provide feature-maps that can be used to produce precise bounding-boxes and estimate the speed of trucks with a high degree of accuracy;
- we showed that the Spatial Correlation Block can improve the accuracy of deep learning speed detection models;
- we demonstrated that the Temporary Stop Mechanism for model training can both improve the accuracy of speed estimation whilst maintaining a strong correlation between the bounding-box and speed;
- we have shown that the proposed model is able maintain a speed estimation loss of less then 5% without camera calibration and where the intrinsic and extrinsic parameters are unknown, and
- we have created a data set that can be used to train vehicle speed detection models from video alone, offering a unique data set.

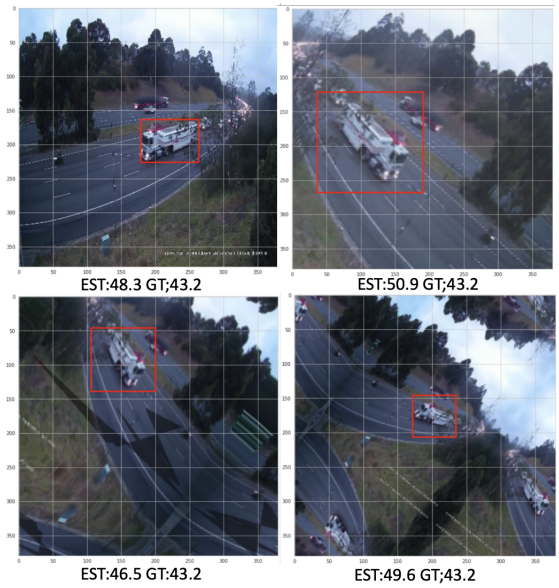

Fig. 7. Demonstration of aggressive augmented images. Top-left is the original image and the others are augmented by different aggressive parameters. EST is the estimated speed and GT is the ground-truth speed.

Although we trained our model with images produced by one camera in one location, we have demonstrated that the model is robust to weather and light conditions, and various transformations such as perspective, shift, scale and rotation. As such, the model can be applied to videos captured by cameras with different setup parameters without necessarily resorting to manual camera calibration. Importantly, our model shows a strong correlation between the predicted speed and the predicted bounding box.

The proposed model and data set have limitations as discussed previously. Importantly, the test data set is produced from a single site. We will continue to work on obtaining more data from additional sites in the future. As more training images are collected from more sites, we believe that the deep learning method should be able to track and estimate the speed of moving vehicles from video capture devices without the need for manual calibration.

We continue to explore use of these models as part of IoT-based solutions for vehicle classification and speed detection services.

ACKNOWLEDGMENT

This research was undertaken using the LIEF HPC-GPGPU Facility hosted at the University of Melbourne. This Fa-

cility was established with the assistance of LIEF Grant LE170100200.

REFERENCES

[1] P.-Y. Sun, W.-Y. Sun, Y. Jin, and R. O. Sinnott, "Heavy vehicle classification through deep learning," in *International Conference on Big Data*. Springer, 2020, pp. 220–236.

[2] Y.-K. Ki and D.-K. Baik, "Model for accurate speed measurement using double-loop detectors," *IEEE Transactions on Vehicular Technology*, vol. 55, no. 4, pp. 1094–1101, 2006.

[3] D. Lhomme-Desages, C. Grand, F. B. Amar, and J.-C. Guinot, "Doppler-based ground speed sensor fusion and slip control for a wheeled rover," *IEEE/ASME Transactions on mechatronics*, vol. 14, no. 4, pp. 484–492, 2009.

[4] H. H. Cheng, B. D. Shaw, J. Palen, B. Lin, B. Chen, and Z. Wang, "Development and field test of a laser-based nonintrusive detection system for identification of vehicles on the highway," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 147–155, 2005.

[5] CEOS, "Tirtl." [Online]. Available: https://www.ceos.com.au/products/tirtl/

[6] S. Javadi, M. Dahl, and M. I. Pettersson, "Vehicle speed measurement model for video-based systems," *Computers & Electrical Engineering*, vol. 76, pp. 238–248, 2019.

[7] S. Hua, M. Kapoor, and D. C. Anastasiu, "Vehicle tracking and speed estimation from traffic videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 153–160.

[8] G. Cheng, Y. Guo, X. Cheng, D. Wang, and J. Zhao, "Real-time detection of vehicle speed based on video image," in *2020 12th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*. IEEE, 2020, pp. 313–317.

[9] H.-Y. Lin, K.-J. Li, and C.-H. Chang, "Vehicle speed detection from a single motion blurred image," *Image and Vision Computing*, vol. 26, no. 10, pp. 1327–1337, 2008.

[10] A. G. Rad, A. Dehghani, and M. R. Karim, "Vehicle speed detection in video image sequences using cvs method," *International Journal of Physical Sciences*, vol. 5, no. 17, pp. 2555–2563, 2010.

[11] F. Yamazaki, W. Liu, and T. T. Vu, "Vehicle extraction and speed detection from digital aerial images," in *IGARSS 2008-2008 IEEE International Geoscience and Remote Sensing Symposium*, vol. 3. IEEE, 2008, pp. III–1334.

[12] J. Sochor, R. Juránek, and A. Herout, "Traffic surveillance camera calibration by 3d model bounding box alignment for accurate vehicle speed measurement," *Computer Vision and Image Understanding*, vol. 161, pp. 87–98, 2017.

[13] M. Dubská, A. Herout, and J. Sochor, "Automatic camera calibration for traffic understanding." in *BMVC*, vol. 4, no. 6, 2014, p. 8.

[14] M. Dubská, A. Herout, R. Juránek, and J. Sochor, "Fully automatic roadside camera calibration for traffic surveillance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1162–1171, 2014.

[15] K. Wang, H. Huang, Y. Li, and F.-Y. Wang, "Research on lane-marking line based camera calibration," in *2007 IEEE International Conference on Vehicular Electronics and Safety*. IEEE, 2007, pp. 1–6.

[16] D. Bell, W. Xiao, and P. James, "Accurate vehicle speed estimation from monocular camera footage," in *XXIV ISPRS Congress*. Newcastle University, 2020.

[17] J. Barros and L. Oliveira, "Deep speed estimation from synthetic and monocular data," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 668–673.

[18] D. C. Luvizon, B. T. Nassu, and R. Minetto, "A video-based system for vehicle speed measurement in urban roadways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1393–1404, 2016.

[19] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6668–6677.

[20] Y. Li, X. Tian, X. Shen, and D. Tao, "Classification and representation joint learning via deep networks." in *IJCAI*, vol. 2017, 2017, p. 67.

[21] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "Siamrpn++: Evolution of siamese visual tracking with very deep networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4282–4291.

[22] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers, and T. Brox, "Flownet: Learning optical flow with convolutional networks," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 2758–2766.

[23] N. Bhandary, C. MacKay, A. Richards, J. Tong, and D. C. Anastasiu, "Robust classification of city roadway objects for traffic related applications," in *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*. IEEE, 2017, pp. 1–6.

[24] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263–7271.

[25] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*. IEEE, 2016, pp. 3464–3468.

[26] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[27] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2462–2470.

[28] R. Cipolla, T. Drummond, and D. P. Robertson, "Camera calibration from vanishing points in image of architectural scenes." in *BMVC*, vol. 99, 1999, pp. 382–391.

[29] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *International journal of computer vision*, vol. 4, no. 2, pp. 127–139, 1990.

[30] R. Dong, B. Li, and Q.-M. Chen, "An automatic calibration method for ptz camera in expressway monitoring system," in *2009 WRI World Congress on Computer Science and Information Engineering*, vol. 6. IEEE, 2009, pp. 636–640.

[31] G. S. K. Fung, N. H. C. Yung, and G. K. Pang, "Camera calibration from road lane markings," *Optical Engineering*, vol. 42, no. 10, pp. 2967–2977, 2003.

[32] Y. Zheng and S. Peng, "A practical roadside camera calibration method based on least squares optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 2, pp. 831–843, 2013.

[33] K.-T. Song and J.-C. Tai, "Dynamic calibration of pan–tilt–zoom cameras for traffic monitoring," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 5, pp. 1091–1103, 2006.

[34] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.

[35] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and flexible image augmentations," *Information*, vol. 11, no. 2, 2020. [Online]. Available: https://www.mdpi.com/2078-2489/11/2/125