

Real-time vehicle counting in complex scene for traffic flow estimation using multi-level convolutional neural network

Zulaikha Kadim^{*}, Khairunnisa Mohd. Johari, Den Fairol, Yuen Shang Li and Hock Woon Hon

Advanced Informatics Lab, MIMOS Berhad, Technology Park, Malaysia

Received: 11-October-2020; Revised: 16-February-2021; Accepted: 17-February-2021

©2021 Zulaikha Kadim et al. This is an open access article distributed under the Creative Commons Attribution (CC BY) License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Abstract

Accurate traffic data collection is crucial to the relevant authorities in ensuring the planning, design, and management of the road network can be done appropriately. Traditionally, traffic data collection was done manually by having human observers at the site to count the vehicle as it passes the observation point. This approach is cost-effective; however, the accuracy can't be verified and may cause danger to the observers. Another common approach is utilizing sensors that need to be installed underneath the road surface to collect traffic data. The accuracy of the data reading from the sensor is highly dependent on the sensor installation, calibration, and reliability which usually deteriorated over time. For these reasons, vision-based approaches have become more popular in traffic flow estimation tasks. Nevertheless, conventional image processing techniques which utilize background subtraction-based approach may face problems in complex highway environment where the number of the vehicle is high, a large gap in vehicle sizes of different classes and high occlusion rate. Thus, in this paper, a real-time vehicle counting in a complex scene for traffic flow estimation using a multi-level convolutional neural network is proposed. By exploiting the capabilities of deep-learning models in delineating and classifying objects in an image, it is shown that the system can achieve average counting accuracy of 97.53% and a weighted average of counting with classification accuracy of 91.5% validated on 585 minutes of highway videos collected from four different cameras; viewing at different vehicle's angles. The system is also capable of running in real-time.

Keywords

Complex urban, Convolutional neural network, Traffic flow estimation, Vehicle counting and classification.

1.Introduction

Due to the rising traffic congestions, information about the traffic conditions can be used by traffic management centres in many ways, including to synchronize traffic lights, to assist drivers in the route's selection, and to assist governments in intercity connections and new roads planning. Traffic management not only provides benefits to the road users, but also to the municipals and central governments, and the environment. Drivers could benefit with less time spent in travelling in urban and rural road, while governments can acquire those data for analysis and used those data to improve urban road management. From the perspective on caring for the environment, this effort could reduce the emission of pollutants.

Conventional techniques for estimating traffic flow, such as deploying the inductive loops, sonar or microwave detectors to obtain traffic flow information, have their disadvantages such as high installation cost, traffic disruption during installation or maintenance, and the failure to detect slow or static vehicles [1].

The recent sensor and communication technology advancement allow traffic management centres as well as municipals to monitor closely the conditions of the urban and rural roads, through developing systems for monitoring the traffic flow, and collecting data on the traffic flow characteristics [2].

In recent year, vision-based systems and image-based sensors have been heavily deployed to collect on-the-road data. These traffic videos are expected to provide on-the-road traffic condition including monitoring abnormalities such as accidents by taking

^{*}Author for correspondence

advantages of the device installed on-site. In comparison, vision-based systems, is easy to install and can be easily upgraded, as the system could be redesigned and its functionalities improved. These days, these vision-based systems are deployed to count and classify vehicles, measure the speed of vehicles, and identify of traffic incidents [3].

Recently, many vision-based techniques are being deployed in intelligent transportation systems to provide traffic information which could derived important clue for many intelligent traffic monitoring systems, such as traffic index and traffic density. This information could then be used to control, optimize and manage traffic [4]. In 2018, Maqbool et al. [5] proposed a simple and effective pipeline to count vehicles, using, at first, a background subtraction technique to detect moving vehicles, and then a blob tracker to track multiple vehicles to avoid duplicated counting for vehicles. In 2016, Liu et al. [6] improved this simple pipeline to be able to work in the compressed domain for videos obtained in highway. In 2016, Xia et al. [7] used the expectation-maximization (EM) algorithm to improve the segmentation quality of moving vehicles. In Prakash et al. [8] used pattern matching to identify traffic density for efficient traffic light management.

These earlier researches, in the context of intelligent transportation, were deployed to detect, track and count vehicles in videos. In earlier papers, vehicles are detected 1) using the model-based methods which uses prior knowledge [9, 10], 2) using the deformable templates for matching targets against known vehicle models in input videos [11], and 3) using simpler features such as corners and edges [12]. As for vehicle tracking, approaches such as mean-shift [13], Kalman filtering [14], and particle filtering [15] were deployed.

The success of deploying CNN-based techniques to detect objects such as using Faster RCNN [16], SSD [17] and YOLO [18], has provide many researcher reasons to deploy these methods to re-look in these methods could be deployed not only to estimate traffic flow [19], but also to other ITS related applications such as road marker and potholes detection [20]. The urban traffic monitoring systems uniqueness, where vehicles may appear to be 1) blurred due to vehicle speed, and 2) small due to occluded vehicles, requires careful consideration when deploying CNN-based techniques to estimate traffic. For that reasons, we have set the objective of this paper is to develop a multi-level convolutional

neural network (mCNN) framework to estimate traffic in complex urban settings, i.e. able to handle various illumination conditions and complication mounting scenario. Herewith, the uniqueness of urban environment in acquisition and processing due to the instrumentation set-up could showcase the proposed mCNN framework for real-time deployment. This framework includes five major modules: 1) pre-processing; 2) object detection; 3) tracking; 4) object classification, and 4) quantification. To handle irrelevant image details, the image is first cropped. The first deep CNN network is then utilized to segment and pre-classified. Later, these pre-classified regions are tracked along the cropped area. Then, the second-deep CNN network is deployed to refine the classification before a quantification module is deployed to deliver the counting results. To test the proposed framework, the quantitative results on realistic videos obtained from different acquisition set-up well demonstrate the performance of the proposed method. The main contributions of this study can be summarized into the following:

- An end-to-end systematic solution for vehicle counting in complex urban setting using CNN-based techniques.
- A novel mCNN framework which could process acquired videos in real-time for vehicles from different views such as frontal or rear.
- A robust mCNN framework which could estimate traffic, i.e. count and track vehicles, in real-time.

The rest of the paper is organized as follows. Section 2 will discuss recent researches related to vehicle counting and classification. Section 3 will then introduce the details of the proposed multi-level convolutional neural network framework. Followed in Section 4, implementation details are described with extensive experimental results on the complex urban setting acquired and the discussion on the results will be provided in Section 5. Finally, we conclude the paper with some future work in Section 6.

2.Literature review

In this section, a number of recent vehicles counting algorithms will be discussed. In Tian et al. [21] presented a general architecture of a traffic analytics for ITS services in four hierarchical layers. The first layer is corresponding to the acquisition layer where visual sensors are used to capture images of traffic scenes. Dynamic and static attributes of the vehicles are then extracted from these images in the second layer by going through detection, tracking and

recognition processes. Detection is the process to locate vehicles in the image. The tracking will then track these detected vehicles to further extract their trajectories information to ensure that each of the vehicles contributes only once to the final counter in

vehicle behaviour understanding layer. Based on this architecture, we have summarized a number of counting algorithms as shown in *Table 1*.

Table 1 Summary of researches in vehicle counting

| Research paper | Dynamic and static vehicle attributes extraction | | | Behavior understanding (counting) |
|----------------------|---|---|--|-----------------------------------|
| | Detection | Tracking | Recognition (classification) | |
| Abdelwahab [22] | Gaussian Mixture Model (GMM) background subtraction, erosion & dilation –narrow ROI | Overlap area between blob in current and previous frame | - | ROI |
| Memon et al. [23] | GMM background subtraction | | Contour Comparison (CC), Bag of Features (BoF) and Support Vector Machine (SVM) method | Line |
| Oltean et al. [24] | YOLOv3-tiny (car, truck and bus) | Motion Estimation | - | ROI |
| Lin and Sun [25] | YOLO (car, bus, and truck) | Shortest distance | - | ROI |
| Santos et al. [26] | YOLOv3 | DeepSort | - | |
| Ciampi et al. [27] | Mask R-CNN (car and truck) | Refresh interval | - | ROI |
| Al-Ariny et al. [28] | Mask R-CNN (car, motorcycle, bus or truck) | KLT tracker | - | ROI |
| Le et al. [29] | Mask R-CNN (car, motorcycle, bus or truck) | - | - | ROI |

Generally, algorithms for vehicle counting can be categorized into classical and deep-learning approach based on the detection method applied. Classical approach particularly background subtraction-based vehicle detection is still prevalent in real application as it is very fast and accurate under certain controlled environment. In [22], Abdelwahab applied Gaussian mixture model (GMM) background subtraction to detect vehicle and further applied a series of morphological operations of erosion and dilation to reduce the blob occlusion. The detection was applied within a narrow region of interest (ROI) to further improve the processing time. The tracking was done based on the overlapping analysis between vehicle pixels area in current and previous frames, where they are considered as the same vehicle if the overlap was more than certain threshold value. Similarly, in [23], Memon proposed GMM for vehicle detection and further classified vehicles into different classes using CC and combination of BoF and SVM. Counting was done by using imaginary line crossing method. Both algorithms worked well for minimal occlusion rate as in frontal or top down camera view, however in perspective view where the occlusion rate is much higher, there will be issues in detection part

specially to detect individual vehicles within the occluded motion area. For that reason, in much recent researches, deep learning approach is more preferable.

Deep-learning approach has shown superior performance not only in image classification tasks but also in detection. Most of the counting system proposed in the literature applying pre-trained models which have been extensively trained using large datasets such as ImageNet and COCO. Detection based on deep-learning models requires higher processing power as compared to the classical approach. Mostly, the processing time is much slower and does not fulfil the real-time requirement. Thus in [24], to achieve real-time, YOLOv3-tiny is proposed. However, there is a trade-off between performance and detection accuracy. Although YOLOv3-tiny is fast it is not as accurate as compared to other deep-learning detection models such as YOLOv3 and mask R-CNN with a more complex network. In [25], YOLOv3 was used for detecting 3 different vehicle classes: car, bus, and truck. Motorcycle was excluded from the counting. In [26], YOLOv3 hyperparameter which is the detection confidence level was tested. It

showed that in some datasets, different hyperparameter value gives a different counting precision level. In [27-29], mask R-CNN was implemented. 27. Ciampi et al. [27] only considered cars and trucks for counting, while Al-Ariny et al. [28] and Eduardo [29] considered 4 classes including motorcycle and bus. Despite higher processing time compared to YOLO, mask R-CNN was preferred as it gives segmentation mask as one of the detection outputs. The mask will be further used to differentiate vehicles within the occluded area. In [28], corner points detected in the image is compared to the segmented mask to identify which points belong to each vehicle. These points were then tracked using KLT tracker. In [29] it was reported that cars can be counted with good precision but not the other vehicle classes such as a motorcycle. This issue may be the result of no tracking was implemented, thus the same vehicles may be counted more than once. They also implemented time interval image capture in which some frames were skipped from processing. The approach may reduce the overall processing time but as the counting area in the image was small, by skipping the frames some vehicles may be miss detected eventually cause miss count. In [24-25] simple tracking methods were proposed to suit the counting application which involves only a small processing area in the image. In [24] tracking was done using motion estimation where the predicted position in the current frame was obtained based on previous positions and dynamic average movement. Measurement of lowest distance was then used to associate active tracker with detected vehicles. Similarly, in [25], the shortest distance between the

coordinates of the vehicle in current and previous frames is used to associate detection and trackers. A simpler tracking approach makes the overall algorithm faster. The counting precision is good in less occluded condition. Thus, the positioning of the camera during video capturing is important, to ensure that the occlusion level between vehicles in the image is kept at a minimum level. In [26], even with a more complex tracker such as DeepSort, it was also shown that overall counting precision was lower in the testing videos with many occlusion scenarios. In this work, deep-learning approach is applied in the detection and classification parts to count and classify six different vehicle classes. Combined with a simple tracking algorithm, the overall counting system is able to run in real-time.

3.Methods

This paper proposes a framework based on CNN, as depicted in *Figure 1*, to detect and count vehicles in complex urban settings. It comprises of five major components, including pre-processing, object detection, tracking, object classification, and quantification. The goal of the pre-processing component is to eliminate irrelevant image details. The object detection module segments a set of interesting regions, and these regions will be pre-classified. Later, these objects are tracked along consecutive video frames and classified into different classes before the counting for cars, vans, trucks, large trucks, buses, or motorcycles is done in quantification component.

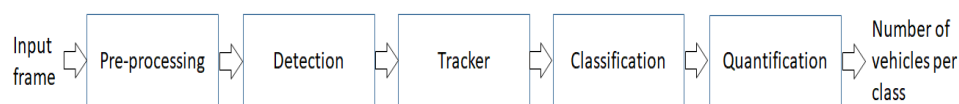


Figure 1 Multi-level CNN (mCNN) Framework

3.1Pre-processing

Due to the instrumentation setup, the presence of some irrelevant details at the edge of the acquired

image sequence can be observed. Thus, a cropping operation is applied (*Figure 2*) to remove these unwanted details.



Figure 1 Sample cropping images for processing (gray area)

3.2 Detection

In this step, a CNN model is applied for detection, i.e. to extract only relevant detected object. These objects consist of four relevant type of vehicle (car, truck, bus, bike or person) with minimum confidence value of 0.6. Two different CNN models are deployed in this detection step, to test the system accuracy and performance. Both models are selected as they can provide superior detection speed as compared to other models such as R-CNN [30] and fast R-CNN [31] with good detection accuracy.

The first model is the SSD Multibox [17]. SSD Multibox which uses a single deep neural network is selected as it could achieve a good balance between the speed performance and accuracy performance, by discarding the fully connected-layers in VGG-16 and replacing with a set of auxiliary convolutional layers, i.e. SSD uses those auxiliary convolutional layers to extract features at multiple scales. These extracted feature maps will be input into a convolutional kernel to predict the bounding box and classification probability. The score of the class probability and the offsets (being the 4 coordinates of a bounding box) is computed. Those scores which exceed the threshold point would be used as the final box to locate the object. Pre-trained SSD model is obtained from [17], which has been trained using COCO dataset [32]. Input image size of the model is 300x300.

The second model is the YOLOv3 [33], which uses a single neural network for both classification and localising the object using bounding boxes. The network has 24 convolutional layers plus 2 fully connected layers. The weights of the first 20 layers in the pretrained convolutional layers are taken from darknet53 which has been trained using Imagenet classification dataset [2, 3]. Then YOLO model is constructed by adding 4 convolutional layers and two fully connected layers and further trained using COCO dataset. YOLO only predict one type of class in each grid. Hence, it would be struggling in detecting a small object or very close object. Pre-trained YOLO model is obtained from [34], which also has been trained using COCO dataset. Input image size of the model is 416x416.

3.3 Tracker

In this step, a tracking algorithm is proposed to track the detected vehicles in successive frames. Tracker is required to make sure that no vehicle is counted more than once. The bounding box information of detected vehicles from vehicle detection step will be used to

initialize the tracker for the first time. In successive frames, detected vehicles information will be used to update the trackers states. In the event of missing observation from detection step, the tracker should be able to predict the position of the vehicle in the image. Thus, a detection model with good detection accuracy will decrease the burden in tracker step and eventually contribute to a higher system accuracy.

The tracker uses Kalman filter with single model (constant velocity) to estimate the vehicle states in each frame. Constant velocity model is used based on the assumption that vehicles within the small road region is moving at constant speed. State vector used is the combination of centroid point, the velocity in x and y directions respectively (v_x and v_y), and the width and height of the vehicle box (w and h). In the event of missing observation, Kalman filter is responsible to predict the possible location of the vehicle respectively.

Track association is completed using the global nearest neighbour based on centroid distance and histogram-based appearance similarity. Tracker trajectories will be associated with the detected vehicles based on the lowest centroid and similarity distances.

It is important to terminate and remove the tracker at the end, as the tracker will consume some processing time and contribute to the overall system performance. Tracker is terminated based on the following few conditions; (1) tracked object is out of region-of-interest; (2) predicted tracked object box is near or beyond image boundary; (3) tracked object already counted and exist for more than few seconds after being counted; (4) vehicle disappears (miss detected) for a number of consecutive frames.

Figure 3 shows sample images with tracker output information. The green boxes are the output from vehicle detection; while blue and red are the output from tracker and quantification respectively. Tracker label for each tracked vehicle is written on the top right of the bounding box. Vehicle trajectories (up to previous 15 frames) are drawn to show the consistency of the tracker output. Note that even during occlusion, i.e. situation whereby few vehicles are close to each other, the tracker could still be able to track vehicles correctly to some extent (e.g. frame # 1255).



Figure 3 Sample tracking output; green boxes show the detected vehicles, while blue and red boxes indicate tracked and counted vehicles respectively. Vehicle will be counted as they pass the line-of-detection (indicated by horizontal black line). The black lines indicate the tracker trajectories in 15 most recent frames, while the number displayed on top of the box is the tracker label

3.4 Classification

For classification process, we employed Inception v1 model which was released in 2014 by Google Inc. It is also known as GoogleNet. This network has won the ILSVRC (ImageNet Large Scale Visual Recognition Competition) in 2014 outperformed other deep learning networks including ZFNet[35], AlexNet[36] and VGGNet[37]. Inception network architecture has 22 layers and it also has inception blocks in which convolution with different kernel sizes is done in parallel and the feature map output is concatenated before pass to the next layer. As there exist large gap in vehicle sizes of different classes (e.g. motorcycle vs. lorry with 3 or more axles), thus, having different kernel sizes as in inception block might be an advantage to us. Our inception model is constructed by utilizing the pretrained network weights from [38] which has been trained on

ImageNet datasets, followed by a new linear classifier layer which consists of a linear and a softmax layer with six output correspond to the six vehicle classes as shown in *Figure 4*. The linear classifier is then trained using our own datasets, while the other network weights are freed. The input image is the RGB images with 224×224 dimension. The data size used during the training is 38352 (90% for the training) and 4262 (10% for validation) respectively.

The challenges in classification include: (1) interclass correlation is high between class 1 and class 2, and (2) intra class correlation is low for class 3 and class 4, especially for vehicle of different viewpoints (frontal and rear).







| Classification | | |
|---|--|--|
| 1. Passenger cars  | 2. Pickups and vans  | 3. 2-axes trucks  |
| 4. 3 or more axes trucks  | 5. Buses  | 6. Motorcycles  |

Figure 4 Vehicle classes

3.5 Quantification

To perform counting, a virtual line is used to trigger the counter to update. As the vehicle crosses the line in the intended direction, the corresponding vehicle class counter will be increased. Sample snapshot with overlaid virtual counting line and the direction is shown in *Figure 5*. In every frame, each of the vehicle centroid will be examined. If the y-coordinate of the centroid is more than the y-coordinate of the virtual line (assuming the origin of the image is on the top-left of the image) and the vehicle is not yet

counted, then the corresponding class counter will be increased by one. Counted vehicle will be flagged as “already counted” to avoid double counting on the same vehicle.

For each tracked vehicle, the history of vehicle class prediction will be kept for a small window frame. The final decision on final vehicle class to update the counting will be regulated based on the frequency of the predicted class of the tracked vehicle.



Figure 5 Sample virtual counting line and the direction

4. Results

4.1 Experimental Setup

Data collection was done using RGB colour video camera, operating with frame rate of 25fps and spatial resolution (standard) of 704×576 , storing 24-bit/pixel, mounted on either a lamp post (P) or a pedestrian flyover (F). The camera set-up for the experiments is illustrated in *Figure 6*. Proposed camera height is more than 5 meters from the ground with proposed camera angle is between 45 to 60 degree from the vertical position. This is to ensure that tall vehicles such as bus and long truck are fully visible in the video, as well as to reduce occlusion level between nearby vehicles. Sample images of video sequences acquired from different highways are included in *Figure 7*. Collected data is then annotated manually by experts in image processing.

Summary of these videos properties is specified in *Table 2*.

After the set-up, we noticed good video sequences were obtained with sufficient spatial resolution for the algorithm to work with but with several challenges: (i) distortion of some video sections by a “shadow effect” caused by the sun –set and –rise scenario, (ii) image noises appearing at the images corners due to acquisition set-up and highway structures, (iii) severe occlusion of vehicles especially during peak hours, (iv) motion blur on fast moving vehicle as shown in *Figure 8*.

On the other hand, initial observation on different vehicle orientation captured from different settings are as follows: 1) for frontal (View 2 and View 4), less occlusion can be observed, but some vehicles from different classes appear almost the same, for example small multi-purpose van (MPV) in class 1 looks the same as big MPVs from class 2, 2) for side view (View 1 and View 3), it is easier to differentiate different vehicles especially the trucks with 2 or more axles, however occlusion rate between nearby vehicles is much higher and become severe during peak hours.

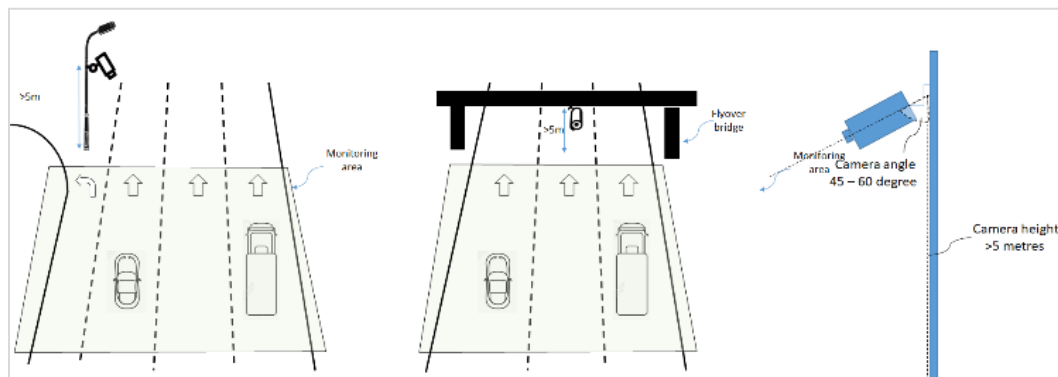


Figure 6 Data collection setup



Figure 7 Camera views

Table 2 Summary of collected data

| Views | Number of lanes | Vehicle orientation | Number of clips (15min/clip) | Average number of vehicles per clip |
|-------|-----------------|---------------------|------------------------------|-------------------------------------|
| 1 | 4 | Side (~15°) | 12 | 1134 |
| 2 | 4 | Frontal | 12 | 1165 |
| 3 | 5 | Side (~45°) | 7 | 2295 |
| 4 | 5 | Frontal | 8 | 1593 |

**Figure 8** Some challenges observed in the datasets; (a) shadow effects caused by the sun –set –rise and severe occlusion, (b) motion blur on fast moving vehicle

4.2 Performance metrics and Results

Proposed system is evaluated in terms of (a) vehicle detection rate and accuracy for different detection models; (b) counting accuracy; (c) counting with classification accuracy; and (d) processing time.

4.2.1 Detection rate and accuracy

Detection rate and accuracy are calculated by comparing the overlapping region between detected boxes and the ground truth (GT). For this evaluation, an intercept over union (IoU) of 50% is considered. If a vehicle with the IoU between its detected box and the ground truth is more than 50%, a true positive (TP) is considered. If the IoU is less than 50%, it is considered as a false negative (FN) case. While false positive (FP) is considered if the system detects non-vehicle object as vehicle. In the event where one vehicle is detected with more than one boxes, the first box is considered as one TP, while the rest are considered as FP.

The detection accuracy (Det_{acc}) and detection rate (DR) are then calculated as follows:

$$Accuracy, Det_{acc} = \frac{TP}{TP+FP+FN} * 100\% \quad (1)$$

$$Detection\ rate, DR = \frac{TP}{TP+FN} * 100\% \quad (2)$$

For evaluating detection accuracy and detection rate, three 20-seconds video clips (~500 image frames) are selected from three camera views. As highlighted in *Table 3*, these videos are carefully chosen to emphasize different problems occurring in complex traffic scene, such as (1) high variance of object size (e.g. motorcycle vs lorry with more than 2 axles) (2) high occlusion between nearby objects (e.g. during peak hour); and (3) different vehicle orientations (e.g. frontal vs side view). For side view, the angle given is the angle between vehicle's major axis and the vertical line. The threshold for minimum confidence value is set to 0.3 for both detection models. These datasets have been manually labelled whereby the vehicle type and bounding box location of each vehicles appears at least 80% of the body within the detection area is recorded as the ground truth. *Table 4* shows the detection accuracy and detection rate of the proposed detection models on the 3 selected video clips. While *Table 5* shows the processing time for both detection models running on Intel® Core™ i5-8400 CPU@2.80GHz processor with Nvidia GeForce RTX 2070 graphic card.

Table 3 Testing video properties

| Dataset (40sec videos) | Total vehicles | Min and max object size (%) | Occlusion level | Orientation |
|------------------------|----------------|-----------------------------|-----------------|-------------|
| Video 1 | 1522 | 0.37, 17.86 | Medium | Side (~15°) |
| Video 2 | 1454 | 0.23, 18.45 | Low | Frontal |
| Video 3 | 5015 | 0.09, 12.45 | High | Side (~45°) |

Table 4 Detection rate and accuracy result on selected 40 seconds videos

| Dataset (40sec videos) | Models | TP | FP | FN | DET_{ACC} (IOU = 50) | DR(%) |
|---------------------------|--------|------|----|------|------------------------|--------------|
| Video 1 | SSD | 996 | 11 | 456 | 68.12 | 68.64 |
| | YOLO | 1161 | 0 | 296 | 79.64 | 79.64 |
| Video 2 | SSD | 668 | 10 | 865 | 43.29 | 43.57 |
| | YOLO | 1139 | 7 | 383 | 74.49 | 74.84 |
| Video 3 | SSD | 3266 | 24 | 1750 | 64.80 | 65.11 |
| | YOLO | 4801 | 68 | 214 | 94.45 | 95.73 |

Table 5 Average inference time for SSD and YOLO

| Detection model | Average inference time (ms) | Average inference time (fps) |
|-----------------|-----------------------------|------------------------------|
| SSD | 9.82 | 101.83 |
| YOLO | 17.88 | 55.92 |

4.2.2 Counting accuracy

Accuracy of the traffic counting is evaluated by using equation 3 and 4. First, the error between counting output and the GT for each video clip is calculated as in equation 3 and 4. Counting is evaluated per video clip instead of per each frame. As an example, if a video A has total vehicles of 100 for the duration of 15mins, and the system reports total vehicle of 85, thus the percentage of error is 15%, and this shall contributes to the accuracy of 85%.

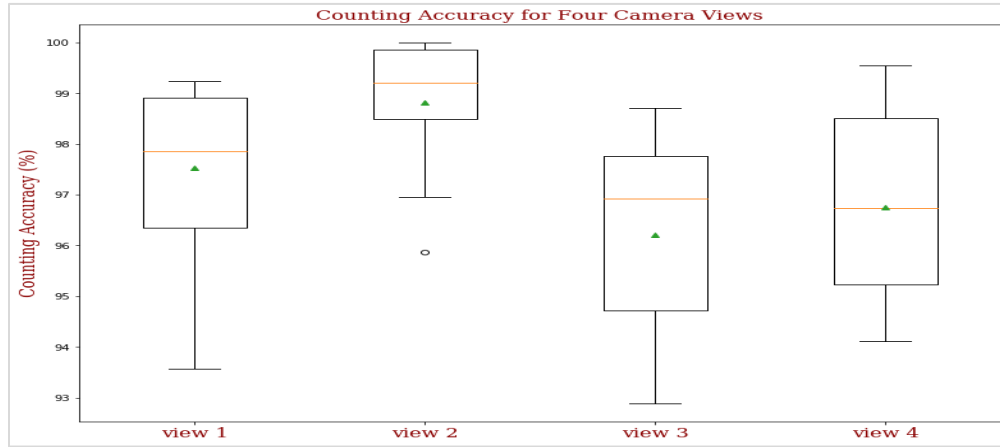
Counting error,

$$Err_c = |GT - Actual| / \max(GT, Actual) \quad (3)$$

Counting accuracy,

$$Acc_c = (1 - E_{tc}) \times 100\% \quad (4)$$

Figure 9 shows the box plot of the accuracy from all four views.

**Figure 9** Box plot of counting results from four camera views

4.2.3 Counting with classification accuracy

For traffic counting with classification, the overall accuracy is calculated using equation (5), where it is the weighted average of the traffic counting accuracy for each class calculated using equations (4). The weight is determined based on the ratio of GT number of each vehicle class to total vehicle in the video clip as in equation (6).

Counting with classification accuracy,

$$Acc_{cc} = \sum_{i=1}^6 w_{ci} * Acc_{ci} \quad (5)$$

Vehicle class x weight,

$$w_{cx} = \frac{GT_{cx}}{\sum_{i=1}^6 GT_{ci}} \quad (6)$$

Figure 10 to Figure 13 depict the accuracy of each class (bar chart) and the weighted average accuracy (line chart) for each camera view respectively.

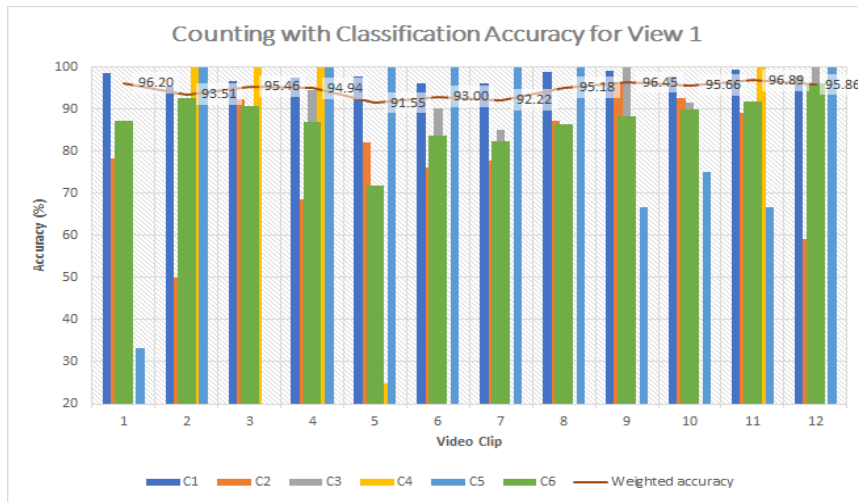


Figure 10 Weighted average accuracy for View 1

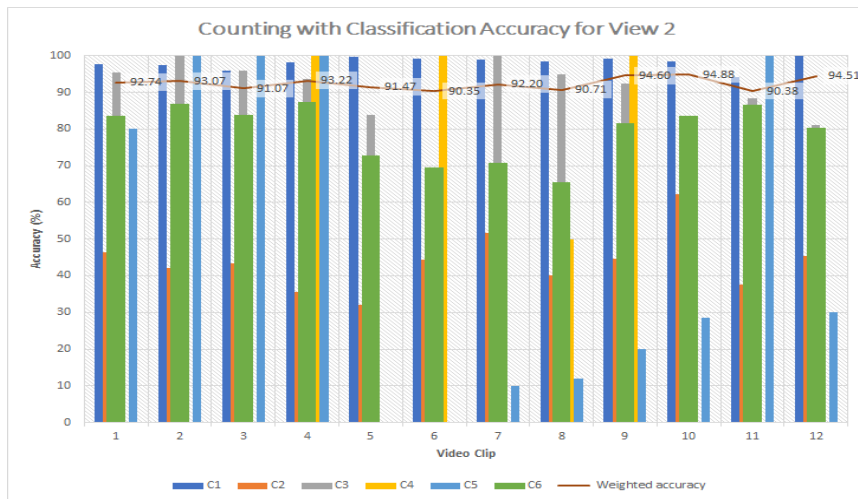


Figure 11 Weighted average accuracy for View 2

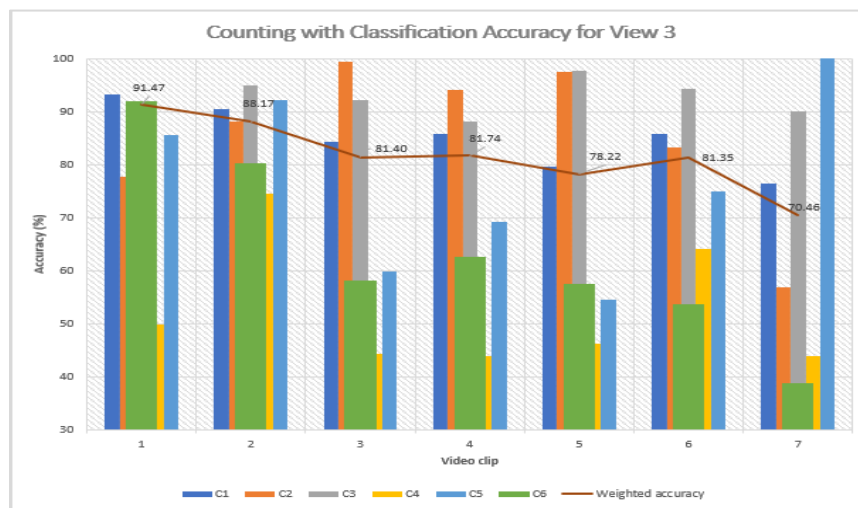


Figure 12 Weighted average accuracy for View 3

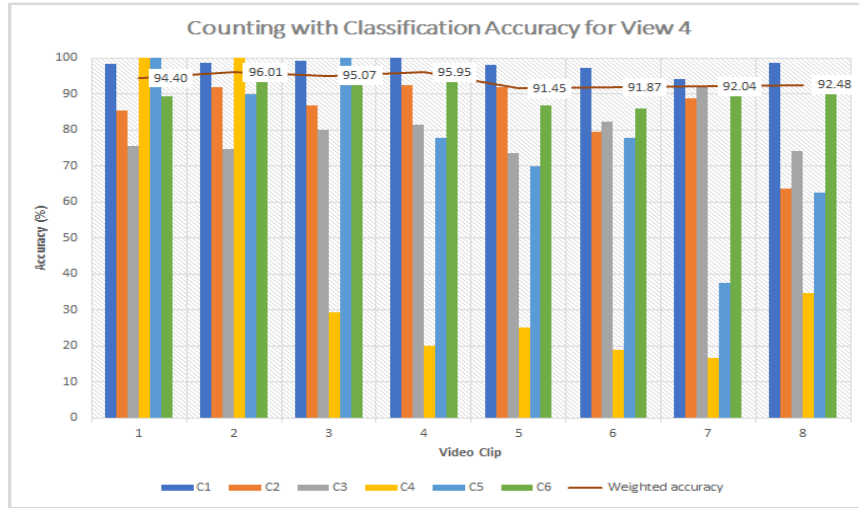


Figure 13 Weighted average accuracy for View 4

4.3 Overall processing time

The algorithm is written using C++, with TensorRT [39] framework is used as the inference engine. All model was converted to TensorRT optimized models for deployment. *Table 6* shows the average processing time of each process. Overall, the system

achieves real-time processing with average of 26.98 frame per second (fps), which is higher than the frame rate of video capturing process at 25fps. Detection step requires longer processing time compared to other processes.

Table 6 Average processing time (per frame)

| Detection (ms) | Tracker (ms) | Classification (ms) | Quantification (ms) | Overall processing | |
|----------------|--------------|---------------------|---------------------|--------------------|-------|
| | | | | Time (ms) | Fps |
| 17.8 | 9.70 | 9.49 | 0.00094 | 37.06 | 26.98 |

5. Discussion

5.1 Detection rate and accuracy

Referring to *Table 4*, YOLO model consistently achieved the highest accuracy and highest detection rate in all videos. YOLO achieves the highest accuracy for Video 3 despite of occlusion level in the video, may due to the orientation of the vehicle which is about 45° from the vertical line. The vehicle appearance is much more distinguishable from this view as compared to frontal view. This is proved by the lowest accuracy achieved by both models in Video 2.

These samples show that YOLO works great in detecting small and occluded vehicles.

In terms of processing time, both models can perform in real-time with SSD runs faster compared to YOLO

with average inference time of 9.82 ms/frames, while YOLO runs at an average of 17.88 ms/frames as listed in *Table 5*. This is explainable as YOLO network has more layers and number of parameters as compared to SSD. Due to its superior performance in terms of detection accuracy and detection rate and its ability to run in real time, YOLO is chosen as the model for our overall system.

Figure 14 further demonstrates the detection outputs of both models running on sample images taken from Video 3. Green and blue boxes are the detected vehicles inside and outside of detection area respectively. While the red boxes are the miss detected vehicles. From the figure, YOLO is able to detect small objects such as motorcycle and occluded objects better than SSD.

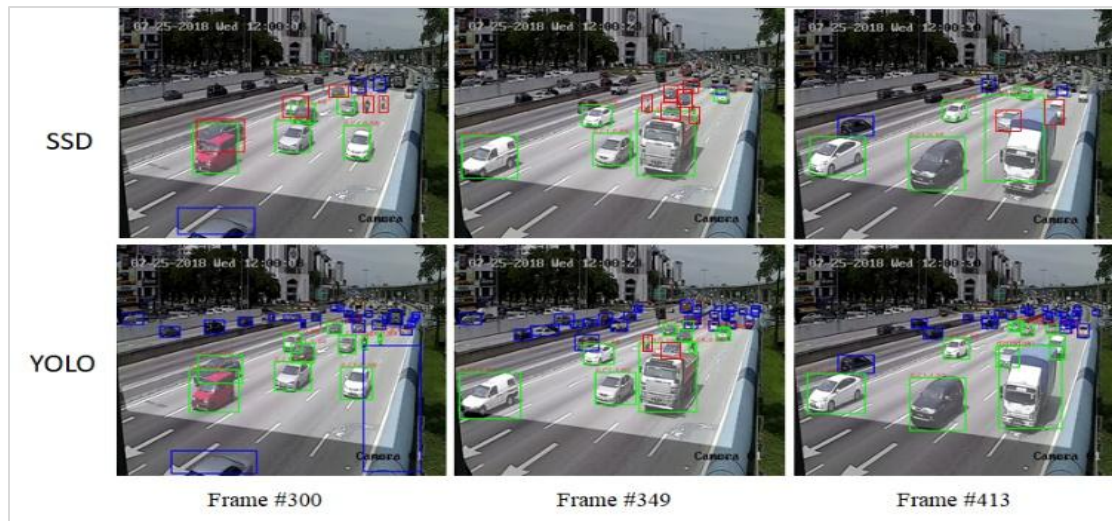


Figure 14 Sample vehicle detection result using different detection models. Detected vehicles within the detection area, detected vehicles outside of detection area and miss detected vehicles are shown in green, blue and red boxes respectively

5.2 Counting accuracy

Based on the results in *Figure 9*, overall, the system has achieved good accuracy results for all video clips, with minimum accuracy of 92.89% is recorded from view 3 for clip between time 1300 to 1315. View 2 achieves the highest mean accuracy, followed by view 1 and view 4, while view 3 records the lowest overall mean accuracy.

Although detection accuracy of samples from view 3 is higher compared to the other views, but the occlusion level is also higher. The occlusion may not impact much on the detection process, however, the tracker may have problem in associating tracker trajectories with the detected vehicles when they are closed and occluded to each other. That gives impact to the overall counting accuracy. Nevertheless, it is also observed that overall counting accuracy is high despite of low detection rate, and this shows that the tracker is able to maintain same vehicle identity during missing observation from detection step.

5.3 Counting with classification accuracy

Each of the counting results is further evaluated according to their vehicle classification output. All camera views except view 3 record weighted average accuracy above 90% for all video clips. However, view 3 records lower weighted average accuracy in general, with the lowest is logged at 70.48% for the clip between 1600-1615 hours. By looking at *Figure*, lower overall accuracy is contributed by the low-class accuracy in C2, C4 and C6.

5.4 Limitation of the proposed method

Results discussed earlier prove that the proposed method is able to count number of vehicles and classify them to six different classes on complex traffic scenes at good accuracy level. However, the system has yet to be tested in more challenging weather and night environment where the lighting condition is low and the image may be distorted with the light beam from the vehicles.

6. Conclusion and future work

A practical framework is proposed in this paper to deploy a real-time vehicle counting system, using (1) an end-to-end systematic traffic flow estimation system in urban setting; (2) a multi-level CNN that could handle vehicles from different camera views; and (3) various datasets to evaluate proposed vehicle counting algorithm in terms of accuracy and performance. Experimental results show that the mCNN could track and count vehicles under various conditions, achieving average counting accuracy and average weighted counting with classification accuracy of 97.53% and 91.5% respectively for all 39 video clips of duration 15-mins each.

From the detection evaluation, it is shown that YOLO outperformed SSD in all sample videos with the highest accuracy is recorded for vehicles from side view (about 45degree angle between vehicle's major axis and vertical line) as the vehicle appearance is more distinguishable from this view. However, occlusion level is also higher in this view. The occlusion level impacts the tracker's performance,

which cause the overall counting accuracy of this view is lower than the others.

In addition to accuracy, real time processing is also a crucial factor in the system deployment and it is proved that our mCNN implementation executes in real-time as the average processes is faster than 25 fps of camera capturing rate with average overall processing of our system is 37.06ms per image, which equivalent to 26.98fps.

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

- [1] Mandellos NA, Keramitsoglou I, Kiranoudis CT. A background subtraction algorithm for detecting and tracking vehicles. *Expert Systems with Applications*. 2011; 38(3):1619-31.
- [2] Cho SY, Quek C, Seah SX, Chong CH. HebbR2-Traffic: a novel application of neuro-fuzzy network for visual based traffic monitoring system. *Expert Systems with Applications*. 2009; 36(3):6343-56.
- [3] Liu Y, Tian B, Chen S, Zhu F, Wang K. A survey of vision-based vehicle detection and tracking techniques in ITS. In *proceedings of IEEE international conference on vehicular electronics and safety* 2013 (pp. 72-7). IEEE.
- [4] Zha ZJ, Wang M, Zheng YT, Yang Y, Hong R, Chua TS. Interactive video indexing with statistical active learning. *IEEE Transactions on Multimedia*. 2011; 14(1):17-27.
- [5] Maqbool S, Khan M, Tahir J, Jalil A, Ali A, Ahmad J. Vehicle detection, tracking and counting. In *international conference on signal and image processing* 2018 (pp. 126-32). IEEE.
- [6] Liu X, Wang Z, Feng J, Xi H. Highway vehicle counting in compressed domain. In *proceedings of the IEEE conference on computer vision and pattern recognition* 2016 (pp. 3016-24).
- [7] Xia Y, Shi X, Song G, Geng Q, Liu Y. Towards improving quality of video-based vehicle counting method for traffic flow estimation. *Signal Processing*. 2016; 120:672-81.
- [8] Prakash UE, Vishnupriya KT, Thankappan A, Balakrishnan AA. Density based traffic control system using image processing. In *international conference on emerging trends and innovations in engineering and technological research* 2018 (pp. 1-4). IEEE.
- [9] Lai JC, Huang SS, Tseng CC. Image-based vehicle tracking and classification on the highway. In *international conference on green circuits and systems* 2010 (pp. 666-70). IEEE.
- [10] Shen Y. The research and applications of visual tracking based on mean shift (Ph. D. thesis). Jiangsu University of Science and Technology. 2008.
- [11] Takeuchi A, Mita S, McAllester D. On-road vehicle tracking using deformable object model and particle filter with integrated likelihoods. In *intelligent vehicles symposium* 2010 (pp. 1014-21). IEEE.
- [12] Tu Q, Xu Y, Zhou M. Robust vehicle tracking based on scale invariant feature transform. In *international conference on information and automation* 2008 (pp. 86-90). IEEE.
- [13] Bouttefroy PL, Bouzerdoum A, Phung SL, Beghdadi A. Vehicle tracking by non-drifting mean-shift using projective kalman filter. In *international ieee conference on intelligent transportation systems* 2008 (pp. 61-6). IEEE.
- [14] Xie L, Zhu G, Wang Y, Xu H, Zhang Z. Real-time vehicles tracking based on Kalman filter in a video-based ITS. In *proceedings. International conference on communications, circuits and systems* 2005(pp. 883-6) IEEE.
- [15] Scharcanski J, de Oliveira AB, Cavalcanti PG, Yari Y. A particle-filtering approach for vehicular tracking adaptive to occlusions. *IEEE Transactions on Vehicular Technology*. 2010; 60(2):381-9.
- [16] Ren S, He K, Girshick R, Sun J. Faster r-cnn: towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*. 2015.
- [17] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg AC. Ssd: Single shot multibox detector. In *European conference on computer vision* 2016 (pp. 21-37). Springer, Cham.
- [18] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In *proceedings of the conference on computer vision and pattern recognition* 2016 (pp. 779-88). IEEE.
- [19] Kadim Z, Johari KM, Samaon DF, Li YS, Hon HW. Real-time deep-learning based traffic volume count for high-traffic urban arterial roads. In *10th symposium on computer applications & industrial electronics* 2020 (pp. 53-8). IEEE.
- [20] Zulkifli N, Laili MA, Saadon SN, Ahmad A, Zabidi A, Yassin IM, et al. Fast region convolutional neural network lane and road defect detection for autonomous vehicle application. *International Journal of Advanced Trends in Computer Science and Engineering*. 2019; 8(1.3):371-80.
- [21] Tian B, Morris BT, Tang M, Liu Y, Yao Y, Gou C, Shen D, Tang S. Hierarchical and networked vehicle surveillance in ITS: a survey. *Transactions on Intelligent Transportation Systems*. 2014; 16(2):557-80.
- [22] Abdelwahab MA. Fast approach for efficient vehicle counting. *Electronics Letters*. 2018; 55(1):20-2.
- [23] Memon S, Bhatti S, Thebo LA, Talpur MM, Memon MA. A video based vehicle detection, counting and classification system. *International Journal of Image, Graphics and Signal Processing*. 2018; 11(9):34.
- [24] Oltean G, Florea C, Orghidan R, Oltean V. Towards real time vehicle counting using yolo-tiny and fast

motion estimation. In international symposium for design and technology in electronic packaging 2019 (pp. 240-3). IEEE.

- [25] Lin JP, Sun MT. A YOLO-based traffic counting system. In conference on technologies and applications of artificial intelligence 2018 (pp. 82-5). IEEE.
- [26] Santos AM, Bastos-Filho CJ, Maciel AM, Lima E. Counting vehicle with high-precision in Brazilian roads using YOLOv3 and deep SORT. In 33rd SIBGRAPI conference on graphics, patterns and images 2020 (pp. 69-76). IEEE.
- [27] Ciampi L, Amato G, Falchi F, Gennaro C, Rabitti F. Counting vehicles with cameras. In SEBD 2018.
- [28] Al-Ariny Z, Abdelwahab MA, Fakhry M, Hasaneen ES. An efficient vehicle counting method using mask R-CNN. In international conference on innovative trends in communication and computer engineering 2020 (pp. 232-7). IEEE.
- [29] Le TT, Aying K, Pama FK, Tabale I. Vehicle count system based on time interval image capture method and deep learning mask R-CNN. In TENCON 2019-2019 IEEE region 10 conference 2019 (pp. 2675-9). IEEE.
- [30] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In proceedings of the IEEE conference on computer vision and pattern recognition 2014 (pp. 580-7).
- [31] Girshick R. Fast R-CNN. In proceedings of the IEEE international conference on computer vision 2015 (pp. 1440-8).
- [32] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft coco: Common objects in context. In European conference on computer vision 2014 (pp. 740-55). Springer, Cham.
- [33] Redmon J, Farhadi A. Yolov3: an incremental improvement. arXiv preprint arXiv:1804.02767. 2018.
- [34] <https://pjreddie.com/media/files/YOLOv3.weights>. Accessed 30 July 2020.
- [35] Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In European conference on computer vision 2014 (pp. 818-33). Springer, Cham.
- [36] Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems. 2012; 25:1097-105.
- [37] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. 2014.
- [38] https://www.dropbox.com/sh/axnbpdl0e92aoyd/AADpmuFIJTxs7zkL_LZrROLa?dl=0. Accessed 30 Mar 2020.
- [39] <https://developer.nvidia.com/tensorrt>. Accessed 30 July 2020.



Zulaikha Kadim received her degree and master degree in Engineering from Multimedia University Malaysia. Currently she is pursuing her PhD in Computer Systems Engineering at Malaysia National University (UKM). She is currently a senior researcher in MIMOS Berhad. Her research interests include Object Detection & Tracking, and Video Analytics. Email: zulaikha.kadim@mimos.my



Khairunnisa Mohamed Johari received her B.Science (Intelligent System) from Mara University of Technology (UiTM) in 2010. She is currently a senior researcher in Advanced Informatics Lab, MIMOS Berhad. Her current research interest are in Video Analytics, Deep Learning and Behavioral Analysis. Email: nisa.johari@mimos.my



Den Fairol Samaon received his B.Science (Computer Science) from University of Technology Malaysia(UTM) in 2003. He is currently a researcher in Advanced Informatics Lab, MIMOS Berhad. His current research interest are in Computer Vision, Deep Learning and Object Detection. Email: fairol.samaon@mimos.my



Yuen Shang Li received Bachelor of Science in Electrical and Electronic Engineering in 2009 and Master Degree of Engineering in Electrical Engineering, majored in Automation and Robotics System in 2011. Currently he is working as research engineer in national R&D institution. His current research interests are in Deep Learning, Video Analytic, Action Recognition and License Plate Recognition. Email: sl.yuen@mimos.my



Hon Hock Woon received his bachelor degree in Electrical and Electronic Engineering from the Nottingham Trent University, United Kingdom in 1997 and his doctorate degree in the same university in year 2000. His main research areas are in imaging/image processing, computer vision and machine learning. Dr. Hon currently holds a position of senior principal researcher at Mimos Berhad. Email: hockwoon.hon@mimos.my

© 2021. This work is published under
<http://creativecommons.org/licenses/by/4.0/> (the “License”). Notwithstanding
the ProQuest Terms and Conditions, you may use this content in accordance
with the terms of the License.