# Urban Traffic Density Estimation Based on Ultra High Resolution UAV Video and Deep Neural Network

Jiasong Zhu, Ke Sun, Sen Jia, Qingquan Li, Xianxu Hou, Weidong Lin, Bozhi Liu and Guoping Qiu

***Abstract*—This paper presents an advanced urban traffic density estimation solution using the latest deep learning techniques to intelligently process ultra high resolution traffic videos taken from an Unmanned Aerial Vehicle (UAV). We first capture nearly an hour-long ultra high resolution traffic video at 5 busy road intersections of a modern megacity by flying an UAV during the rush hours. We then randomly sampled over 17K 512x512 pixel image patches from the video frames and manually annotated over 64K vehicles to form a dataset for this research which will also be made available to the research community for research purposes. Our innovative urban traffics analysis solution consists of advanced deep neural network based vehicle detection and localization, type (car, bus and truck) recognition, tracking and vehicle counting over time. We will present extensive experimental results to demonstrate the effectiveness of our solution. We will show that our enhanced Single Shot Multi-box Detector (Enhanced-SSD) outperforms other deep neural network based techniques and that deep learning techniques are more effective than traditional computer vision techniques in traffic video analysis. We will also show that ultra high resolution video provides more information which enables more accurate vehicle detection and recognition than lower resolution contents. This paper not only demonstrates the advantages of using the latest technological advancements (ultra high resolution video and UAV) but also provides an advanced deep neural network based solution for exploiting these technological advancements for urban traffic density estimation.**

***Index Terms*—Unmanned aerial vehicle (UAV), road traffic monitoring, traffic density estimation, deep neural networks, vehicle detection, vehicle tracking, vehicle counting.**

## I. Introduction

URBAN traffic monitoring has long been a popular research topic among scholars and industrial practitioners. Considering the rapid growth of the metropolis road network and the booming of cars in recent decades, it is indispensable to build a more comprehensive system to help understand the intricate transportation system in the urban area. Conventional traffic monitoring systems rely on thousands of detectors (e.g. cameras, induction loops, radar sensors) deployed on fixed locations with small detecting ranges to help capture various road conditions throughout the network [1]–[4]. Such kind of systems have exhibited many limitations in terms of range and effectiveness. For instance, if the information is required beyond the scope of these fixed detectors (i.e. blind regions), human labors are then frequently deployed to assess these particular road conditions [5]. Besides, many monitoring tasks require to temporally detect detailed traffic conditions such as sources and destinations of the traffic flow, regions of incidents and queuing information at crossroads [6]–[8]. To achieve this, the visual information of multiple fixed detectors need to be aggregated in order to provide a relatively large view of the interested area, which could introduce extra noisy information and the overhead costs. Therefore, it is essential to develop a more effective approach for acquiring visual information.

To tackle these issues, some previous work attempt to exploit still satellite images for traffic monitoring [9]–[12]. Satellite images are good to capture still scenes like land usage and mineral deposits inspection, however, due to the hysteretic nature (the data are captured several days or months ago), satellite images cannot capture the real-time or recent traffic conditions and they also they lack spatial resolution for specific ground locations. Besides, the high overhead cost and environmental interferences such as bad weather and air pollution also hinder its application to the traffic monitoring area.

Recent developments in low air-borne Unmanned Aerial Vehicle (UAV) platforms, sensor, and image processing techniques have resulted in an increasing research interests of this technology in the remote sensing science community. Lots of studies have successfully demonstrated UAV operations using small platforms equipped with sensors for RGB, multispectral, hyper-spectral, and thermal imaging, as well as with laser scanning capabilities [13]–[16]. Compared with satellite images, the UAV based remote sensing platforms have shown obvious advantages: the location and height of the UAV is easy to change, and the targets of monitoring can be a large area or a few specific moving objects, and video collection and processing can be near-real time [17]–[20]. Anyway, the UAV's capacity for near-continuous acquisition of ultra-high

This work was jointly supported in part by the National Natural Science Foundation of China under Grant 61773414, and in part by the Shenzhen Future Industry Development Funding program under Grant 201607281039561400, and the Shenzhen Scientific Research and Development Funding Program under Grant JCYJ20170818092931604 *(Corresponding author: Guoping Qiu.)*

J. Zhu, K. Sun, Q. Li and W. Lin are with the Shenzhen Key Laboratory of Spatial Information Smarting Sensing and Services, Shenzhen University, China (e-mail: zhujiasong@gmail.com, sk100.force@gmail.com, liqq@szu.edu.cn, linwaydong@163.com).

S. Jia is with the Computer Vision Research Institute, College of Computer Science and Software Engineering, Shenzhen University, China, and also with the Shenzhen Key Laboratory of Spatial Information Smarting Sensing and Services, Shenzhen University, China (e-mail: senjia@szu.edu.cn).

X. Hou, B. Liu and G. Qiu are with Guangdong Key Laboratory of Intelligent Information Processing, College of Information Engineering, Shenzhen University, China. G. Qiu is also with the School of Computer Science, The University of Nottingham, UK (e-mail: hxianxu@gmail.com, lucifer.bozhi@gmail.com, qiu@szu.edu.cn).

resolution imagery has provided both opportunities and challenges in the area of traffic monitoring and management. There are significant research opportunities in detecting running vehicles, identifying vehicle types, estimating traffic load and discovering traffic accidents and disaster forecasting [21]–[24]. Hence, novel UAV-based remote sensing imaging techniques and the development of specialized processing workflows (e.g. deep learning techniques) will allow us to address novel and important science questions in the field of city traffic monitoring.

Among these science questions, traffic density estimation is of significant importance since it provides direct information about the traffic condition in various locations across the city road network, helping traffic authority better design traffic rules and manage the light signal system [20], [21], [23]. However, UAVs are not widely applied in the traffic density estimation system due to specific challenges for detecting and tracking vehicles in the high resolution UAV's images and videos.

On one hand, the equipped camera of a UAV may rotate and shift during the recoding process. On the other hand, compared with conventional monitoring systems, the UAV's video contains not only the ordinary data such as the global view of the traffic flow, but also each vehicle's own data like its moving trajectory, lane changing information and interaction with other vehicles [25], [26]. Therefore, the UAV's video needs to be recorded using a very high resolution and frame frequency so as to capture adequate ground details. This inevitably leads to a huge size of the UAV's video data and pose challenges for vehicle detection and tracking algorithms [27]. Besides, UAV videos also contain various background information (noise) and poses significant challenges to the traditional video analysis approaches.

### A. Related Work

*1) Detection Based Approaches:* Many existing vehicle detection methods adopts sliding window based searching and hand-crafted feature matching techniques to identify and localize vehicles in an image (or a frame in videos) [5], [28]–[30], however, due to the lack of high-level semantic information in terms of vehicle types, all the detected objects are treated as vehicles. Note that identifying vehicle types in traffic density estimation is essential since the size and capacity of different vehicles have different impact on the traffic pressure. Some work adopts extra classifiers to recognize different types of vehicles [31], but their approaches introduce more overhead cost for computation and parameter optimization.

*2) Motion Based Approaches:* Several methods try to estimate traffic density using motion based vehicle tracking techniques (e.g. background subtraction and optical flow) [5], [32], [33]. These approaches could work well on simple traffic scenes such as expressway and roads in the rural area, but they tend to fail in the urban traffic scene due to the distraction of various background information and intricate local ground conditions. Additionally, some vehicles appear in only a few frames and their trajectories cannot be accurately estimated.

*3) Deep learning based Approaches:* Recently a few deep learning based methods were proposed for object density estimation [34]–[37]. These methods attempt to predict the object density from the holistic view using deep neural networks (DNNs). However, the original images have to be downsampled in order to be processed by DNN, which would lead to the loss of local pixel-wise information. Besides, the problem of scale variation of moving objects is not well addressed.

To summarize, detection based and motion based approaches cannot work well on city road traffic density estimation, because they are sensitive to video quality and road conditions. Moreover, many existing methods are unable to provide the accurate number of different types of vehicles.

### B. Our Contributions

In order to deeply understand city road traffic density and overcome the challenges brought by the real world UAV video data, we develop a robust Deep Vehicle Counting Framework (DVCF) which is capable of counting different types of vehicles in high resolution videos. To the best of our knowledge, this is the first framework which integrates deep neural networks and traditional algorithms for analyzing high resolution ($3840 \times 2178$) UAV road traffic videos.

*1) Deep Vehicle Counting Framework:* The DVCF contains two main parts: the first part is deep learning based vehicle detection with type identification, and the other part is vehicle tracking and counting. For vehicle detection and type identification, we train an enhanced Single Shot Multi-box Detector [38] (Enhanced-SSD) in which we replace the backbone VGG [39] model to more powerful ResNet [40] model and then redesign all the feature layers to further improve the detection performance. The proposed enhanced-SSD further strengthens the ability of detecting vehicles in various types (i.e. cars, buses and trucks, see Fig. 1) and sizes, and its superiority has been demonstrated in the experiments.

In the vehicle tracking and counting part, we detect vehicles frame-by-frame for an input video, then develop a simple online tracking algorithm to associate detections to unique identities (with types) across the whole video sequence. The number of vehicles can be obtained by measuring the number of these unique identities. Many previous traditional object tracking algorithms are unable to keep the types of tracked objects during the tracking progress, and we have solved this issue and make them support multiple objects with multiple types.

*2) UAV City Traffic Video Dataset:* We have collected and labeled a large-scale UAV city traffic video dataset (UavCT) from 5 busy intersections of the city. It contains 101,970 frames (56m 39s) with a $3840 \times 2178$ resolution. Such a high resolution can capture a large view of the traffic area, and also provides more ground details than videos recorded with a low-resolution. To make this dataset more challenging, we intentionally collect the video data during peak hours. In the UavCT, we define three common vehicle types: car, bus and truck. Unlike existing car dataset such as KITTI [41] and UA-DETRAC [42] which mainly focus on specific vehicle

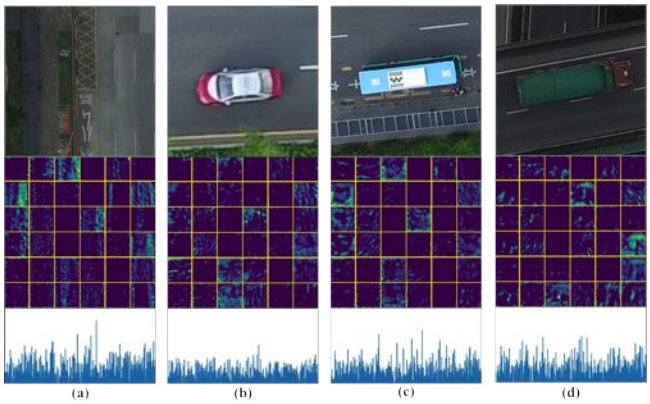



Fig. 1. Example of deep features computed on different types of vehicles including the background: (a) background, (b) car, (c) bus and (d) truck. In the top row, the original images are shown, while related deep features extracted from a convolution layer and a fully-connect layer are illustrated in the middle and bottom row respectively.

models, our dataset emphasizes on real world road traffic flow analysis in metropolis. We hope this novel dataset is beneficial for motivating research in vision based traffic flow analysis. We will make this dataset publicly available for academic purposes.

The rest of paper is organized as follows. Section 2 introduces the traffic data acquisition and pre-processing. Section 3 elaborates the Deep Vehicle Counting Framework. Section 4 presents experiment settings, results and discussion. Section 5 concludes the paper and discuss the future work.

## II. Data Acquisition and Details of UavCT

A UAV traffic monitoring system has been set up to acquire traffic video data, which consists of a quadrocopter (Fig. 2a), a remote controller with built-in video transfer system (Fig. 2b) and a camera mount (Fig. 2c).

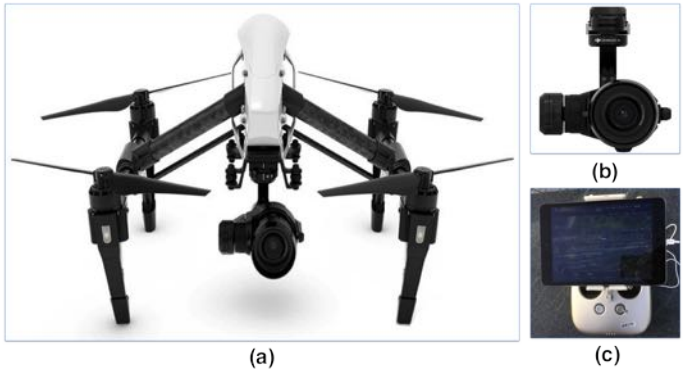



Fig. 2. The UAV traffic monitoring system used in this paper: (a) the whole set, (b) the camera mount and (c) the remote controller .

### A. Data Acquisition

The quadrocopter used in the experiments is the DJI Inspire 1 Pro. It contains motors, main controller, battery and the connection port for the camera mount. The UAV is designed to be lightweight, flexible and stable when recording high-quality videos. With the help of the built-in inertial measurement unit (IMU) which incorporates both a 6-axis gyroscope and an accelerometer for movement compensation, the camera mount mounted by the UAV is capable of stably recording road traffic at $3840 \times 2178$ resolution (30fps). The third part of the UAV is the remote controller which transmits real time video stream and UAVs' flight data back to the controller, such as the distance between the aircraft and the remote controller, GPS location, flight velocity, etc. We collected traffic data in Shenzhen, a typical metropolis which undertakes significant traffic pressure in China. We pick five key road intersections in Shenzhen to acquire our traffic videos. Considering the capability issues, the videos were firstly stored in the camera's SD card, and then transferred to the computer. Some parametric settings of the data collection are listed in Table I.

TABLE I
Parameters settings for data collection

| Parameter | Range/Value |
|---|---|
| Time slot | Peak hours (7:00-9:00am, 5:00-7:00pm) |
| Weather condition | Sunny/cloudy |
| Operating temperature | -10°C∼50°C |
| Hovering altitude | 126 meters above ground |
| Hovering accuracy (GPS Mode) | Vertical: 0.5 m, Horizontal: 2.5 m |
| Ground resolution | 5.5 cm per pixel |
| Number of videos taken | 10 |
| Video length of each record | up to 10 minutes |
| Video resolution | $3840 \times 2178$ |
| FPS | 30 |

### B. Dataset Details

In the UavCT, the video's physical resolution is 5.5 cm per pixel at the ground level. This makes vehicles range in size from 80 to 180 pixels. To build the training set, we first temporally subsample the original video frames by a factor of 150, then for each frame in the subset, we divide it into small patches with a uniform size of $512 \times 512$. We allow an overlapping area of 200 pixels vertically and horizontally between these patches to ensure each vehicle appears as a complete object. The final training set contains 17,186 image patches. These patches are then annotated with the following information: (i): *Bounding box*: rectangle surrounding each vehicle. (ii): *Vehicle type*: three general types including car, bus, and truck. Note that we do not define very specific vehicle categories (e.g. private cars, taxi, etc.) because too many types would inevitably exacerbate the problem of unbalanced data, which would lead to sub-optimal performance of machine learning algorithms. An example of data annotation is illustrated in Fig. 3.

For a vehicle appearing on the overlapping area, we annotate it if more than half of its shape appears in that image patch. If a vehicle is not annotated in the current image patch, it is guaranteed to be annotated in an adjacent patch. Besides, we asked annotators to avoid annotating the same vehicle more than once during the annotation progress. However, in the testing phase, they are required to be detected multiple times

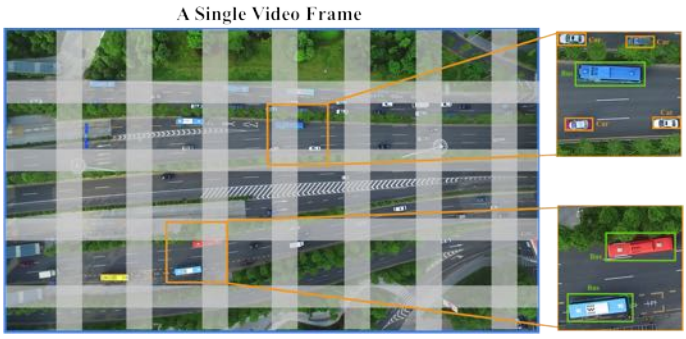



Fig. 3. A example of data annotation performed on the UacCT dataset. The overlapping areas are denoted by gray bars. The yellow boxes are image patches extracted from the original video frame. In each patch, vehicle are annotated with bounding boxes and corresponding types.

at different locations across the whole video sequence. More details about the training are described in Table II.

TABLE II
THE NUMBER OF ANNOTATED VEHICLES IN EACH TRAINING VIDEO OF THE UAVCT DATASET.

| Type/Video | 1 | 2 | 3 | 4 | 5 | Total |
|---|---|---|---|---|---|---|
| Car | 12190 | 9680 | 6281 | 8409 | 15531 | 52091 |
| Bus | 3082 | 940 | 704 | 181 | 1411 | 6318 |
| Truck | 330 | 284 | 143 | 3174 | 1727 | 5658 |
| Total | 15602 | 10904 | 7128 | 11764 | 18669 | 64067 |

For building the testing set, we collect traffic data from the five road intersections again but limit the length of each video to be no more than 100 seconds. That is to say, we take testing videos at different time slot from training videos. We then construct two testing sets to evaluate the proposed framework. The first one contains five images, each image is one full resolution frame which are randomly sampled from each testing video accordingly. The second testing set consists of the original five testing videos (see Fig. 4). The whole length of videos in testing set 2 is 5m 30s. To build the ground truth of the testing set 1, we ask human subjects to count the numbers of different types of vehicles in each image in set 1. For the testing set 2, only vehicles within the road range are considered. More details of testing set 1 and set 2 could be found in Table III and Table IV respectively.

TABLE III
THE NUMBER OF VEHICLES IN EACH IMAGE IN THE TESTING SET 1.

| Type/Image | 1 | 2 | 3 | 4 | 5 | Total |
|---|---|---|---|---|---|---|
| Car | 77 | 60 | 81 | 23 | 42 | 283 |
| Bus | 21 | 3 | 3 | 0 | 8 | 35 |
| Truck | 2 | 0 | 1 | 5 | 6 | 14 |
| Total | 100 | 63 | 85 | 28 | 56 | 332 |

## III. A DEEP VEHICLE COUNTING FRAMEWORK

We handle the vehicle counting problem in two stages. The first stage is a modified sliding window based Single Shot Multi-box Detector (Enhanced-SSD), which is able to produce bounding boxes of vehicles with type information in high resolution videos. The second stage is a fast multi-object tracker applied on these bounding boxes, estimating each vehicle's trajectory, maintaining its unique identity and the corresponding type. Such a tracking-by-detecting framework is a natural approach to process very high resolution UAV videos since performing vehicle detecting and tracking simultaneously is almost impossible considering such high computational cost. The whole workflow is illustrated in Fig. 5.

TABLE IV
THE NUMBER OF VEHICLES IN EACH VIDEO IN THE TESTING SET 2.

| Type/Video | 1 | 2 | 3 | 4 | 5 | Total |
|---|---|---|---|---|---|---|
| Car | 124 | 65 | 90 | 124 | 195 | 598 |
| Bus | 22 | 7 | 4 | 1 | 24 | 58 |
| Truck | 2 | 0 | 5 | 36 | 28 | 71 |
| Total | 148 | 72 | 99 | 161 | 247 | 727 |

### A. Motivation of Deep Learning Based vehicle detection

In this work, vehicle detection is done by feature matching, and good feature representation can help generating good detection results. We notice that deep feature representation which is proposed recently has shown significant superiority over conventional features in multiple fields of computer vision, such as image classification [40], image segmentation [43], object detection and tracking [44]. Hence we are interested that how well the deep features can do compared to conventional features in our experiments. To do this, we design a multi-label image classification test to distinguish vehicles from the background as well as predicting corresponding vehicle types.

*1) Settings:* We train a linear Support Vector Machine (SVM) classifier to identify four categories: car, bus, truck and the background. To build the sub-dataset, we randomly subsample 4,000 images from the training set. Each image appears as a small block which contains at most one object: either a vehicle or just the background (see Fig. 6). We randomly select 3,200 images for training the classifier and remaining images are used for testing. For feature representation, we extract 3 types of deep features from different deep neural network models which are popular for image classification, namely AlexNet [45], VGGNet [39] and ResNet [40]. We then compute 3 famous conventional features, namely Scale Invariant Feature Transform (SIFT) [46], Speeded-up robust features (SURF) [47] and Histograms of Oriented Gradients (HOG) [48] features for comparison. For implementation, we use the Caffe [49] toolkit to extract deep features and use OpenCV [50] for conventional feature extraction and SVM based classification. More details of feature architecture are listed in Table V.

We use the classification accuracy (CA) to evaluate the classification performance. CA is defined as either the fraction or the count of correct predictions. In multi-label classification, if the entire set of predicted labels completely match the ground truth labels, the CA would be 1.0, otherwise it is 0.0.


Fig. 4. Overview of the two testing sets.

TABLE V
ARCHITECTURE OF FEATURES USED IN THE CLASSIFICATION EXPERIMENT.

| Feature | Type | Layer | Dimension |
|---|---|---|---|
| AlexNet | deep | fc7 | 4096 |
| VGGNet | deep | fc7 | 1024 |
| ResNet | deep | pool5 | 2048 |
| SIFT | conventional | – | 12800 |
| SURF | conventional | – | 6400 |
| HOG | conventional | – | 861840 |

Denote $p_i$ as the predicted label of $i$th testing sample and $g_i$ as the corresponding ground truth label, then the classification accuracy could be formulated as:

$$AC(p, g) = \frac{1}{N} \sum_{i=1}^{N} \varphi(p_i, g_i) \tag{1}$$

where $\varphi(p_i, g_i)$ is an indicator function which equals to 1 if $p_i = g_i$, otherwise it is 0. $N$ represents the number of testing samples.

*2) Results and Discussion:* We perform the classification test on the subset containing 4000 images and the quantitative results are shown in Fig. 7. The CA scores of deep features are significantly higher (more than 20%) than the conventional features, and features extracted from ResNet is ranked as the first place over the other five types of features. These are reasonable results since ResNet (101 layers) is much deeper than AlexNet (7 layers) and VGGNet (16 layers), which means it is able to capture more in-depth information in the images, thus yielding relatively higher level feature representation. This is also one reason why we use ResNet as our base network in Enhanced-SSD. Since deep features perform overwhelmingly well in the classification test, we only consider deep learning based approaches in the following experiments.

We also show the training time and the testing time using different types of visual features in Table VI. It can be seen that the time consumption using the VGGNet feature is lowest amongst the six types of features. This is because the feature dimension of VGGNet feature is 1024 and it is significantly lower than other types of features, A low dimension means less computational cost of the classifier. However, classification accuracy (CA) score using ResNet feature is higher than the VGGNet feature, so we still use ResNet as our backbone network in the Enhanced-SSD in consideration of performance.

TABLE VI
THE TRAINING TIME AND THE TESTING TIME (IN SECONDS, LOWER IS BETTER) USING DIFFERENT TYPES OF VISUAL FEATURES. THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Feature type | Training (s) | Testing (s) |
|---|---|---|
| AlexNet | 173.8 | 40.3 |
| VGGNet | **92.3** | **21.8** |
| ResNet | 136.6 | 37.7 |
| SIFT | 296.4 | 65.2 |
| SURF | 216.6 | 48.1 |
| HOG | 2351.5 | 526.4 |

### B. Vehicle Detection by Enhanced-SSD

We propose a deep learning based vehicle detection method. Unlike previous work which perform vehicle detection and classification separately, we integrate these two parts in a whole deep neural network (DNN) named Enhanced-SSD (see Fig. 8).

The two main components of the Enhanced-SSD are feature description and vehicle localization. To generate feature descriptors with class information, a set of convolution layers in a DNN which is initially applied for image classification are used to construct our base network. The main difference between Enhanced-SSD and the conventional SSD [38] is

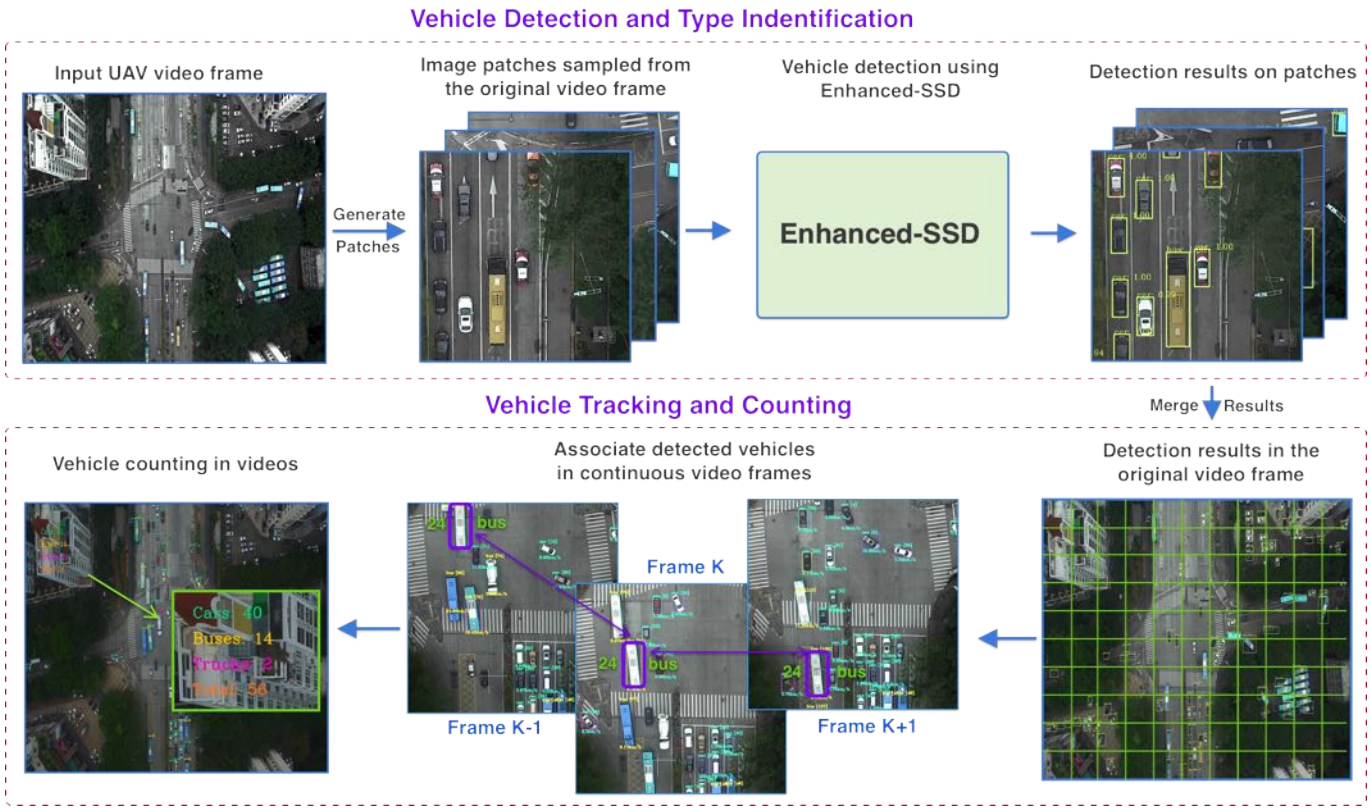



Fig. 5. The working flow of the Deep Vehicle Counting Network. Detection is performed on image patches which are extracted from the original input video frame, and then the results are stitched back together to obtain the global result. In the tracking & counting phase, a set of trackers are built to capture unique vehicle identities across the whole video sequence. The numbers of vehicles could be obtained by counting the outputs of trackers.


Fig. 6. Examples of the sub-dataset for image classification. The *car* class contains private cars, taxis, SUVs and small vans, etc. The *bus* type refers to buses and coaches. While the *truck* category include various trucks and large-sized multi-functional vehicles.

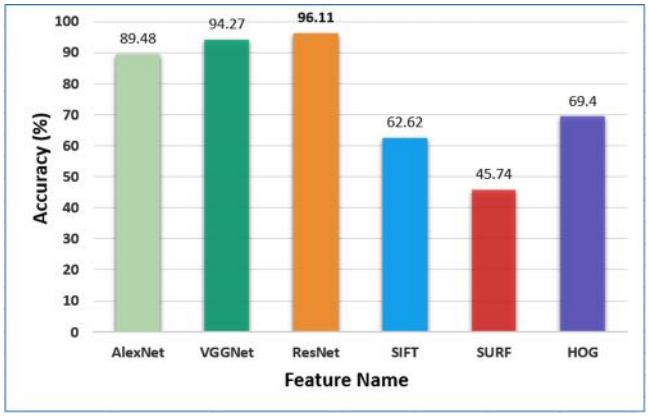



Fig. 7. Results of the vehicle type classification using different types of features. The highest classification accuracy score is highlighted using the bold font.

that we employ the more powerful classification model called ResNet [40] as our based model. We do this because SSD has the limitation that small objects (i.e. small vehicles in high resolution videos) are not detected well, and replacing the original VGGNet [39] with ResNet would increase the number of layers and total number of channels, which could further improve the detection performance. Besides feature description, another key component is vehicle localization. To achieve this, we add several auxiliary convolution layers and a pooling layer to the base network for predicting locations of vehicles. Then we feed the output of 6 layers (two from the reduced ResNet and four from newly added layers) to the classification layer for generating each vehicle's location and its corresponding category. (see Fig. 8). Note that Pool6 is a Global Average Pooling [51] layer. Moreover, only layers in the multi-scale feature layer group are illustrated in Fig. 8. The other three feature layers, namely Conv1-2, Conv2-2 and Conv3-2 are all designed with 512 filters, the kernel size is 3

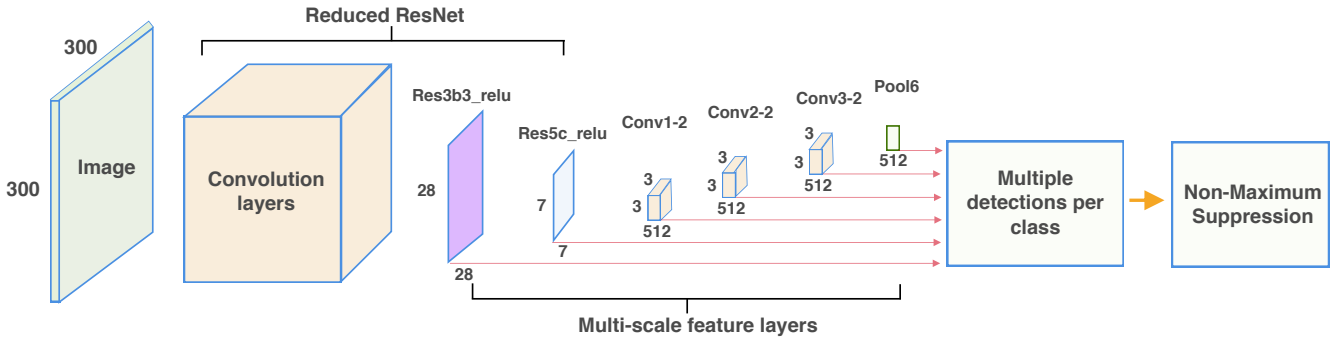



Fig. 8. Overall structure of the Enhanced-SSD. The layers from Res3b3 to Pool6 which are fed into the classifier layer are responsible for predicting the locations of vehicles in different scales and aspect ratios. All outputs of the detected bounding boxes with class confidence scores are filtered by thresholding in the Non-Maximum Suppression algorithm.

$\times$ 3, the padding size is 1 and the stride is 2.

*1) Multi-scale feature layers for vehicle detection:* Similar to the conventional SSD, in a convolutional manner, we randomly initialize a set of detecting boxes with various scales and aspect ratios at each location in these multi-scale feature layers. For each detecting box, we calculate shape offsets [38] and confidences (class scores) for each vehicle's category $\{C_1, C_2, ..., Cn\}$. In a feature layer of size $i \times j$ with $p$ channels, a small kernel (e.g. $4 \times 4$) is applied to predict either the class score or the shape offsets relative to the coordinates of the initial detecting box. More specifically, for each detecting box in $m$ given locations, we compute $n$ class scores and the 4 offsets relative to the original detecting boxes. This yields a total number of $(n+4)m$ filters which are applied on each location in the feature map, and resulting in $(n+4)mij$ outputs for a $m \times n$ feature map (see Fig. 9). Varying the shape of detecting boxes in several feature maps allows us to effectively capture different vehicle sizes in high resolution traffic videos.

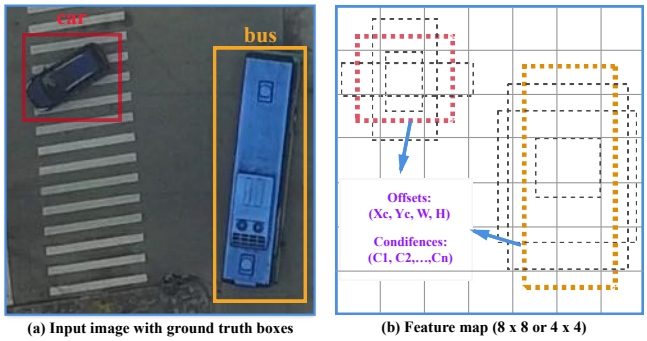



Fig. 9. Structure of the multi-scale feature maps in Enhanced-SSD. Given an input image with ground truth bounding boxes (e.g. (a)), we first initialize a small set of detecting boxes with various aspect ratios at each location in several feature maps (e.g. 8 $\times$ 8 or 4 $\times$ 4 scales in (b)). Then for each detecting box, we predict the shape offsets of bounding boxes and the class scores for all vehicle types.

*2) Training:* Training Enhanced-SSD is straightforward: we find which detecting boxes are close to a ground truth box and then tuning the parameters of the network accordingly. Specifically, for each ground truth box, we firstly pick a small set of detecting boxes with various locations, sizes and aspect ratios. And then we sequentially match each ground truth box to the detecting box according to the best Jaccard overlap [52]. Unlike the approach in [52] which reserves only one detecting box with the maximum overlap, we replace it with a threshold $\alpha$ (0.5 in our experiments) and match detecting boxes to any ground truth box with the Jaccard overlap higher than $\alpha$. This reduces the computational cost and avoids the risk of missing detecting boxes with high Jaccard overlap scores.

The training objective is extended from MultiBox objective [52] by adding support for multiple vehicle categories. It is consisted of two components: the localization loss (loca) and class confidence loss (conf). The former is responsible for localizing vehicles in input images and the latter is aiming to identify their corresponding types. Denoting $\delta(p, g, c) = \{0, 1\}$ as an indicator for matching the detecting box $p$ to ground truth box $g$ with category $c$, then according to the matching strategy, we could have $\sum_i \delta(p, g, c) \geq 1$. The overall objective function is:

$$L(\delta, c, p, g) = \frac{1}{N_{match}}(\gamma L_{loca}(\delta, p, g) + L_{conf}(\delta, c)) \quad (2)$$

where $N_{match}$ is the number of matched detecting boxes, $L_{loca}(\delta, p, g)$ refers to the Smooth $L_1$ [53] localization loss between the ground truth box ($g$) and the predicted box ($p$). We represent the bounding box using its center coordinates, width and height, and apply regression on the offsets of these parameters. The confidence loss $L_{conf}(\delta, c)$ is the softmax loss for multi-class classification. The weight term $\gamma$ controls the proportion of localization loss and is set to 1 in our experiments (validated by cross validation).

During the training phase, the detection algorithm would inherently learn the sizes of different types of vehicles during training. For example, for a bus, the detection algorithm is likely to assign a large rectangle, while for a car the algorithm would assign it with a relatively small rectangle. In the testing phase, several predicted bounding boxes with confidence scores (indicating the vehicle type) would be generated for a single vehicle, and one with the highest confidence score would be reserved as the final prediction for this vehicle. Hence, if a large rectangle is predicted on a car, its confidence score for the car type would be very low, and this prediction is likely to be discarded by the detection algorithm.

*3) Testing:* In the testing phase, an input image is fed into the trained Enhanced-SSD, and couples of predicted boxes with class confidence scores are generated as the initial

output. For each unique vehicle, only a single prediction (bounding box and type) is reserved via thresholding using Non-Maximum Suppression algorithm [54].

One important issue in testing is that the input scale of Enhanced-SSD is $300 \times 300$, hence directly using the original high resolution ($3840 \times 2178$) traffic video frames is not applicable. To fill the scalability gap, we designed a region-based strategy by employing a sliding window to divide the original video frame into small patches with the size of $512 \times 512$. We allow an overlap of 200 pixels horizontally and vertically between patches in order to capture complete vehicles. We then perform detections on each image patch and stitch them back together to the initial scale (see Fig. 5).

Allowing overlaps between these patches could obtain complete detections, however, this also increases the numbers of repeated detections (i.e. a single vehicle is detected multiple times in different patches). To solve this issue, in our experiments, we eliminate repeated boxes by evaluating: either their center distances are smaller than a threshold ($T_{cd}$) or their IoU (intersection over union) scores are above a threshold ($T_{iou}$). IoU is a popular evaluation criterion in the field of object detection [52], [53], which is used to measure the ratio of overlap between two bounding boxes. In our case, the IoU score of two predicted boxes $B_i$ and $B_j$ is:

$$IoU(B_i, B_j) = \frac{B_i \cap B_j}{B_i \cup B_j} \tag{3}$$

$IoU = 1$ represents a complete match between two bounding boxes. After we obtained all repeated boxes on a single vehicle, we reserve the one with the maximum scale.

### C. Vehicle Tracking and Counting

As mentioned in Section III, the proposed DVCF is a tracking-by-detection framework. Since we could obtain detection results in the whole video sequence, we simplify the multiple object tracking (MOT) as a data association problem which is aiming to associate detections across different frames in a video sequence. Hence, we found that traditional algorithms are quite appropriate for this objective in term of accuracy and efficiency. Compared with previous work [55]–[57] which focus on a single variation of objects and low resolution videos, our approach is able to handle multiple types of objects simultaneously in high resolution videos.

In our approach, only the location coordinates of bounding boxes and corresponding vehicle types are considered for motion estimation and data association. Moreover, long-term occlusion is ignored as it occurs infrequently in road traffic videos. Designing vehicle re-identification algorithms maybe helpful to fight this problem, however, it would introduce significant overhead cost to the whole framework, which potentially hinders its usage in real world applications.

*1) Motion Estimation:* To estimate motions for each unique vehicle, we represent it using a linear model and propagate its identity into the next frame. And each modeled vehicle is independent of other vehicles. The state of each vehicle is represented using a column vector:

$$V = [x_c, y_c, s, a, c, \hat{x_c}, \hat{y_c}, \hat{s}]^T \tag{4}$$

where $x_c$ and $y_c$ represents the horizontal and vertical centers of the vehicle bounding box, while $s$ and $a$ refer to its scale and aspect ratio respectively. The vehicle category is denoted as $c$. Note that the aspect ratio and the vehicle category are treated as constant during the tracking progress. Once a detection is assigned to a vehicle, its bounding box is used to update its state via the Kalman filter algorithm [58].

*2) Data Association:* For assigning detections to vehicles over time, each vehicle's motion (bounding box coordinates) is estimated by computing its new location in the current frame. We then create a cost matrix $M_{cost}$ by measuring the intersection-over-union (IoU) between each detection and predicted bounding boxes of the existing vehicles. Hence, our goal is to find an optimal assignment to maximize the numbers of matches in these two sets of bounding boxes. In our experiments, we solve it via the Hungarian algorithm [59]. Again, a threshold $Th_{assign}$ is set to discard assignments with low IoU scores between detections and bounding boxes of existing vehicles.

*3) Life Cycle Management of Tracks:* Track maintenance is an essential aspect of vehicle tracking. When vehicles enter or leave the traffic scene, unique trackers need to be created or deleted accordingly over time. In the first frame, a set of trackers are initialized by measuring locations (bounding box coordinates) of existing vehicles. Then in the following frames, the state of assigned trackers are updated using the matched detections, while any unassigned detection may start a new track. For creating a new tracker, we treat any detection with an overlap (to existing trackers) lower than $T_{assign}$ as an untracked vehicle.

Each track will keep count of a number of consecutive frames, where no new detections are assigned. If this number exceeds a threshold $T_{miss}$, the target is assumed to have left the field of traffic view and the track is terminated. This avoids overgrowing the number of trackers and reducing tracking errors caused by missed detections over a long-term period. In our experiments, we empirically set $T_{miss}$ to 10. We do this because trackers are initialized under the assumption that the velocity of moving targets is constant in short-term tracking, which means that it is a poor indicator to model the true dynamic movements in a long period. Besides, early deletion of missing targets improves efficiency.

*4) Vehicle Counting:* Using the results of tracking, counting vehicles is simple. Each newly created tracker contains an unique ID, the vehicle type, and its bounding box coordinates. The numbers of different types of vehicles could be obtained by inspecting the number of trackers created with the specific type.

## IV. Experiment and Discussion

We thoroughly evaluate our Enhanced-SSD on vehicle counting tasks on these two testing set in comparison with three other deep learning based approaches.

### A. Counting Vehicles in high resolution Images (testing set 1)

In this section, we evaluate the proposed DVCF for vehicle counting in traffic images. Our objective is to count all types of vehicle in all the five images in the testing set 1.

*1) Settings:* The training dataset in the UavCT contains 17,168 image patches. We randomly select 85% of these patches for training and the remaining 15% for validation. We compare our approach (RSSD) with three recent deep learning based object detection methods, including the conventional SSD [38], Faster RCNN (FRC) [60] and YOLO [61]. We train the four deep models using Caffe [49] toolkit on a GTX 1080Ti GPU with 11 GB video memory. The setting of main training parameters is shown in Table VII. We use smaller batch size to train the Enhanced-SSD to avoid the problem of insufficient video memory. The optimizer is set to stochastic gradient descent (SGD) for better performance in this experiment. We initialize the learning rate as 0.001 and it begins to decrease to the one tenth of current value after 20,000 epochs. The momentum is set to 0.9 by default according to these models.

TABLE VII
TRAINING PARAMETERS OF THE FOUR DEEP LEARNING BASED APPROACHES.

| Model | SSD | Faster-RCNN | YOLO | Enhanced-SSD |
|---|---|---|---|---|
| Batch size | 32 | 32 | 32 | 6 |
| Optimizer | SGD | SGD | SGD | SGD |
| Learning rate | 0.001 | 0.001 | 0.001 | 0.001 |
| Momentum | 0.9 | 0.9 | 0.9 | 0.9 |
| Epoch | 12,000 | 12,000 | 60,000 | 12,000 |

In the testing phase, a testing image is divided into small patches ($512 \times 512$) with an overlap of 200 pixels, and these patches are then fed into the trained network to detect vehicles. The final result is obtained by aggregating detection results on all patches. We eliminate the repeated bounding boxes on each vehicle by setting center distance threshold $T_{cd}$ as 0.3 and IoU threshold $T_{iou}$ as 0.1 respectively.

The values of parameters in the experiments are all empirically set because finding theoretically optimal values almost impossible for the proposed models (this is also the case in many machine learning problems). Fortunately, we have enough data and are able to perform k-fold cross-validation to pick appropriate parameter values for the proposed model. In this work, we set k to 10 based on the manual observation of the data and the investigation of previous work using cross-validation. More concretely, for a single parameter, we first set a possible value range for it (e.g. [0.1, 0.5] with a step of 0.05 for the IoU threshold), and then use 10-fold cross-validation to pick the best value. For a group of parameters, we employ grid search strategy to find the best combination of parameter values in the parametric space via 10-fold cross-validation.

To make vehicle counting more straightforward, the detection result is visualized by drawing vehicle locations and corresponding types on the input image. Then counting is done naturally by measuring the number of these bounding boxes. We quantitatively evaluate the counting result via correctness (Cor), completeness (Com) and quality (Qua), which are defined in [62]. The true positives (TP) means the number of correctly detected vehicles, false positives (FP) represents the number of invalid detections and false negatives (FN) denotes the number of missed vehicles. Among the three evaluation criteria, quality is most important since it considers both correctness and completeness of detection algorithms.

$$Correctness = \frac{TP}{TP + FP} \tag{5}$$

$$Completeness = \frac{TP}{TP + FN} \tag{6}$$

$$Quality = \frac{TP}{TP + FP + FN} \tag{7}$$

*2) Results and Discussion:* We report the overall counting result on testing set 1 (see Table VIII) instead of each image because it better describes the overall performance and robustness of these detection algorithms. It is obvious that our Enhanced-SSD achieves the best performance in terms of quality on the testing set 1, followed by the conventional SSD. For correctness, YOLO earns the highest score. However, this method yields too many false negatives (missing vehicles) which leads to very low scores of completeness and quality. Faster-RCNN obtains similar results as YOLO, but with lower scores of correctness.

We also visualize the detection results on testing image 2 as an example to show the overall performance of deep learning based approaches (see Fig. 10). Cars, buses and trucks (if any) are automatically marked with light green, orange and light blue bounding boxes respectively. The small images in the middle are patches extracted from the original images which give clearer ground details for type-specific detection. We have noticed that all four methods except YOLO generates a small number of false positives. That's no accident because in the training set, only regions containing vehicles are annotated by human annotators, while non-vehicle area (including pure background and empty road) are ignored. A few ignored regions may exhibit very similar appearance with particular vehicles (especially buses and trucks) which would consequently lead to a few wrong detections. YOLO does not give false positives probably because it takes a relatively conservative strategy by setting a high threshold to rejects many potentially correct detections in order to avoid generating wrong detections. This is why it misses a large number of vehicles (see orange boxes of YOLO in Fig. 10). Besides, both Enhanced-SSD and the conventional SSD performs very well on this task, but SSD gives more wrong detections and misses more vehicles than our model.

The counting results of specific vehicle types are illustrated in Table IX. We show the counting quality measure here to evaluate the overall performance of these four deep learning approaches. It is obvious that the proposed Enhanced-SSD outperforms the other three methods on all vehicle types (especially on "truck"). This again demonstrates the effectiveness and versatility of our method.

### *B. Counting Vehicle in high resolution Videos (testing set 2)*

Since our DVCF is a tracking-by-detection approach, in this section, we investigate how detection performance will affect vehicle tracking and how well our tracking method could work collaboratively with the deep neural networks. Hence, our objective is counting all types of vehicles in high resolution testing videos.

TABLE VIII
COUNTING RESULTS ON THE TESTING SET 1 (IMAGES). FOR EACH METHOD, WE SHOW THE TP, FP, FN, CORRECTNESS, COMPLETENESS AND QUALITY. THE BEST VALUES ARE HIGHLIGHTED BY BOLD FONTS. UP ARROW MEANS HIGHER IS BETTER, AND DOWN ARROW DENOTES LOWER IS BETTER.

| Method | TP (true positive) ↑ | FP (false positive) ↓ | FN (false negative) ↓ | Correctness ↑ | Completeness ↑ | Quality ↑ |
|---|---|---|---|---|---|---|
| Faster-RCNN | 184 | 58 | 138 | 0.760 | 0.571 | 0.484 |
| YOLO | 158 | **1** | 174 | **0.994** | 0.476 | 0.474 |
| SSD | 287 | 8 | 45 | 0.973 | 0.864 | 0.844 |
| Enhanced-SSD (ours) | **293** | 7 | **39** | 0.977 | **0.883** | **0.864** |


Fig. 10. Counting results on testing image 2 using four deep learning approaches. True positives (correctly detected vehicles) are marked using green boxes with corresponding class types and confidence scores, while false positives (erroneous detections) are denoted using red boxes and false negatives (missed vehicles) are surrounded by orange boxes.

TABLE IX
COUNTING RESULTS OF SPECIFIC VEHICLE TYPES ON TESTING SET 1 (IMAGES). THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Type/Method | Faster-RCNN | YOLO | SSD | Enhanced-SSD |
|---|---|---|---|---|
| Car | 0.486 | 0.521 | 0.864 | **0.866** |
| Bus | 0.674 | 0.229 | 0.829 | **0.892** |
| Truck | 0.25 | 0.143 | 0.571 | **0.80** |

*1) Setting:* Given a testing video, we perform frame-by-frame detection following the temporal order (i.e., from the first frame to the last frame) using the trained network. This significantly challenges the generalization ability and robustness of detection algorithms, because in the training set, most vehicles are annotated only once, and it is then required to be detected multiple times (at different locations) across the whole video sequence.

Once detection is done, A set of trackers are created to associate bounding boxes with different vehicles across the whole video sequence. We empirically set the threshold $T_{assign}$ as 0.3 to start a new track, and set $T_{miss}$ as 10 to terminate a track. However, repeated tracker may still be created for the same vehicle because the algorithm loses track on it (exceeding the $T_{miss}$) and treat it as another new identity. To avoid this, we only initialize trackers according to the detection results of the first frame in testing videos, then we only focus on vehicles who enter the traffic view in the testing videos and build new trackers for them. Another possible solution is to design vehicle re-identification algorithms which also consider visual appearance to associate detections to vehicles, not just bounding boxes. Nevertheless, it would inevitably introduce extra computational cost which leads to reduced efficiency. We will work on efficient and robust vehicle re-identification algorithms in our future work.

During the tracking phase, vehicles which are not within the range of roads (e.g. parking lots) are ignored in the counting phase since they contribute nothing to estimate the city traffic density. We manually define the road ranges since testing videos contain large-range and complex traffic scenes. For example, crossroads, T-junctions, slip roads and ring roads are often intertwined with buildings and planting, and they cannot be accurately identified by current automatic road-detection algorithms. Recent work on semantic segmentation may be helpful but it is beyond the scope of this work. For experiment implementation, we run the tracking algorithm on 5 testing videos using an Intel i7-6700K CPU with 32GB on-board memory.

Because our DVCF separates the vehicle detection and tracking into two different phases, the vehicle counting results should be demonstrated separately. However, there's no common approach or standard to evaluate the result of vehicle counting in videos. In this work, we define the Counting Accuracy (CA) which is based on the "tracking rate" defined in [63] as the evaluation criteria. The CA is formulated as $CA = \frac{N_{sc}}{N_t}$, where $N_{sc}$ refers to the number of successfully counted vehicles, and $N_t$ represents the total number of vehicles in the testing video. Counting of a vehicle is failed if the algorithm does not create a tracker for it during the tracking phase.

TABLE X
COUNTING RESULTS ON THE TESTING SET 2 (VIDEOS). WE LIST THE NUMBER OF SUCCESSFULLY COUNTED VEHICLE ($N_{sc}$), THE TOTAL NUMBER OF VEHICLES ($N_t$) AND THE COUNTING ACCURACY. THE BEST VALUES ARE HIGHLIGHTED USING BOLD FONTS.

| Method | $N_{sc}$ / $N_t$ | Counting Accuracy (CA) |
|---|---|---|
| Faster-RCNN | 470 / 727 | 0.646 |
| YOLO | 427 / 727 | 0.587 |
| SSD | 663 / 727 | 0.912 |
| Enhanced-SSD (ours) | **681 / 727** | **0.937** |

TABLE XI
COUNTING RESULTS OF SPECIFIC VEHICLE TYPES ON TESTING SET 2 (VIDEOS). THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Type / Method | Faster-RCNN | YOLO | SSD | Enhanced-SSD |
|---|---|---|---|---|
| Car | 0.609 | 0.634 | 0.926 | **0.950** |
| Bus | 0.879 | 0.431 | 0.914 | **0.931** |
| Truck | 0.775 | 0.324 | 0.789 | **0.831** |

*2) Results and Discussion:* The overall results of vehicle counting on testing set 2 are listed in Table X, from which we can see that the proposed Enhanced-SSD achieves the best CA score on the testing set. This also indicates that good detection performance would lead to promising results in vehicle tracking and counting. Besides, Faster-RCNN and YOLO do not perform well on this task because they miss many vehicles in the detection phase (see Fig. 10), which undoubtedly poses negative impacts on the counting results. Similar results are obtained when we measure type-specific vehicle counting (see Table XI): on each vehicle type, our Enhanced-SSD outperforms other deep learning based approaches, especially for truck detection.

We also provide the training time and the testing time/speed in Table XII. It can be observed that training and testing of the Faster-RCNN method takes the longest time amongst the four approaches. The YOLO method achieves the lowest training time and the fastest testing speed on the testing set 2, which is due to its shallow network architecture compared with other models. However, its detection performance is significantly worse than other methods. The SSD and Enhanced-SSD (ours) approaches obtained similar results in terms of time consumption, however, the enhanced-SSD performs better than SSD when inspecting the performance (accuracy) of vehicle counting.

### C. The Impact of the Resolution

Although our data was recorded using ultra high resolution, we were interested to determine if a high resolution really helps detection results. To do this, we created an auxiliary set where we down-sampled all the testing images in the testing set 1 and testing set 2. By adjusting the resolution of each image, we can determine performance changes of our detection algorithms. More specifically, we resize each original testing image to a resolution of 2K (2560), 1080p ($1920 \times 1080$), and 720p ($1280 \times 720$) respectively. We then count all types of vehicles in these low-resolution images/videos.

The results are shown in Table XIII and Table XIV. We can see the counting performance degrades dramatically when image resolution goes down. It makes sense because in a high resolution image (both for training set and the testing set), a vehicle generally takes a few pixels. However, in a 720p image, it only takes one or two pixels. This makes these vehicles (especially small cars) totally unrecognizable (see Fig. 11) for vision-based algorithms. Hence, recording data in a high resolution is necessary since it provides enough ground details to help the detection algorithms accurately localizing different types of vehicles.

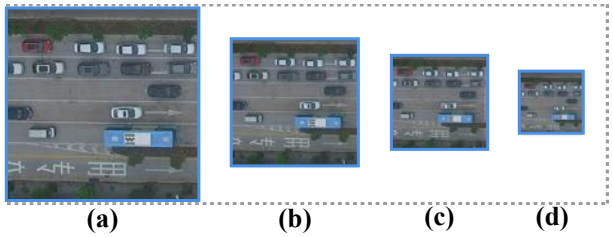



Fig. 11. Example of an image patch in different resolution: (a) 4K, (b) 2K, (c) 1080p and (d) 720p.

We show the testing time/speed of vehicle counting on the testing set 1 (images) and testing set 2 (videos) in different resolutions (see Table XV). It can be seen that the time consumption decreases when the resolution goes down. This is because fewer vehicle are detected and tracked in low resolution images/videos, which requires relatively low computational cost. Anyway, we do not considering using a low resolution in practical applications since the counting performance degrades rapidly when the resolution goes down.

## V. CONCLUDING REMARKS

In this paper, a UAV and deep learning based vehicle detecting, tracking and counting system has been presented with some advantages in the traffic density estimation system. The proposed Deep Vehicle Count Framework (DVCF) effectively and efficiently extracts traffic density data from the high resolution UAV videos at various geo-location with complex traffic view scopes. To summarize, three significant features of our approach have been demonstrated:

(i) A UAV city traffic video dataset is created to help estimate the real-world city traffic density and is also aiming to motivate research in vision based traffic flow analysis in intricate traffic views.

TABLE XII
THE TIME CONSUMPTION OF VEHICLE COUNTING IN TESTING SET 1 (IMAGES) AND TESTING SET 2 (VIDEOS). THE UP ARROW MEANS HIGHER IS BETTER WHILE THE DOWN ARROW DEMOTES LOWER IS BETTER. THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Method | Faster-RCNN | YOLO | SSD | Enhanced-SSD |
|---|---|---|---|---|
| Training time ↓ | 87h 29m | **57h 31m** | 64h 58 m | 66h 42m |
| Testing time on testing set 1 (images) ↓ | 52.35s | 11.15s | **7.95s** | 10.85s |
| Testing speed on testing set 2 (videos) ↑ | 35.3 fps | **72.3 fps** | 44.1 fps | 46.2 fps |

TABLE XIII
COUNTING RESULTS (QUALITY MEASURE) OF SPECIFIC VEHICLE TYPES ON TESTING SET 1 (IMAGES) WITH DIFFERENT RESOLUTIONS. THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Type / Resolution | 720P | 1080P | 2K | 4K |
|---|---|---|---|---|
| Car | 0.056 | 0.096 | 0.256 | **0.886** |
| Bus | 0.045 | 0.112 | 0.288 | **0.892** |
| Truck | 0.043 | 0.104 | 0.349 | **0.80** |

TABLE XIV
COUNTING RESULTS (COUNTING ACCURACY MEASURE) OF SPECIFIC VEHICLE TYPES ON TESTING SET 2 (VIDEOS) WITH DIFFERENT RESOLUTIONS. THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Type / Resolution | 720P | 1080P | 2K | 4K |
|---|---|---|---|---|
| Car | 0.074 | 0.127 | 0.397 | **0.950** |
| Bus | 0.063 | 0.124 | 0.361 | **0.931** |
| Truck | 0.065 | 0.111 | 0.329 | **0.831** |

(ii) In this work, the DVCF is specifically designed for vehicle detection, classification, tracking and counting. However, it is easy to be extended to detect and track many other types of objects (e.g. people, bicycles etc.).

(iii) The proposed DVCF presents a successful attempt to integrate conventional vision based algorithms and deep learning based approaches. Compared with recent methods, our approach considers both the accuracy and the efficiency, while exhibiting good robustness.

For future work, one can think of developing more effective vehicle detection and tracking algorithms while achieving a high processing speed and robustness. Besides, another direction is designing a method to automatically select wanted regions (city roads) to further reduce human supervision and improve the overall efficiency.

TABLE XV
THE TIME CONSUMPTION OF VEHICLE COUNTING ON THE TESTING SET 1 (IMAGES) AND TESTING SET 2 (VIDEOS) IN DIFFERENT RESOLUTIONS. THE UP ARROW MEANS HIGHER IS BETTER AND THE DOWNARROW DENOTES LOWER IS BETTER. THE BEST VALUES ARE HIGHLIGHTED USING A BOLD TYPEFACE.

| Method | 720P | 1080P | 2K | 4K |
|---|---|---|---|---|
| Testing time on set 1 ↓ | **2.95s** | 7.5s | 9.4s | 10.85s |
| Testing speed on set 2 ↑ | **125.3fps** | 104.1fps | 78.2fps | 46.2fps |

## REFERENCES


[1] Z. Kim and J. Malik, "Fast vehicle detection with probabilistic feature grouping and its application to vehicle tracking," in *null*. IEEE, 2003, p. 524.

[2] B. T. Morris and M. M. Trivedi, "A survey of vision-based trajectory learning and analysis for surveillance," *IEEE transactions on circuits and systems for video technology*, vol. 18, no. 8, pp. 1114–1127, 2008.

[3] R. Cucchiara, M. Piccardi, and P. Mello, "Image analysis and rule-based reasoning for a traffic monitoring system," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 119–130, 2000.

[4] A. Yoneyama, C.-H. Yeh, and C.-C. J. Kuo, "Robust vehicle and traffic information extraction for highway surveillance," *EURASIP Journal on Applied Signal Processing*, vol. 2005, pp. 2305–2321, 2005.

[5] L. Wang, F. Chen, and H. Yin, "Detecting and tracking vehicles in traffic by unmanned aerial vehicles," *Automation in construction*, vol. 72, pp. 294–308, 2016.

[6] C. Brooks, R. Dobson, D. Banach, D. Dean, T. Oommen, R. Wolf, T. Havens, T. Ahlborn, and B. Hart, "Evaluating the use of unmanned aerial vehicles for transportation purposes," *MDOT, Houghton, Michigan, MTRI-MDOTUAV-FR-2014*, 2015.

[7] H. Zhou, H. Kong, L. Wei, D. C. Creighton, and S. Nahavandi, "Efficient road detection and tracking for unmanned aerial vehicle," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 297–309, 2015.

[8] H. Oh, S. Kim, H.-S. Shin, A. Tsourdos, and B. A. White, "Behaviour recognition of ground vehicle using airborne monitoring of unmanned aerial vehicles," *International Journal of Systems Science*, vol. 45, no. 12, pp. 2499–2514, 2014.

[9] J. Leitloff, S. Hinz, and U. Stilla, "Vehicle detection in very high resolution satellite images of city areas," *IEEE transactions on Geoscience and remote sensing*, vol. 48, no. 7, pp. 2795–2806, 2010.

[10] S. Hinz, "Detection and counting of cars in aerial images," in *Image Processing, 2003. ICIP 2003. Proceedings. 2003 International Conference on*, vol. 3. IEEE, 2003, pp. III–997.

[11] W. Yao and U. Stilla, "Comparison of two methods for vehicle extraction from airborne lidar data toward motion analysis," *IEEE Geoscience and remote sensing letters*, vol. 8, no. 4, pp. 607–611, 2011.

[12] T. N. Mundhenk, G. Konjevod, W. A. Sakla, and K. Boakye, "A large contextual dataset for classification, detection and counting of cars with deep learning," in *European Conference on Computer Vision*. Springer, 2016, pp. 785–800.

[13] C. Romero-Trigueros, P. A. Nortes, J. J. Alarcón, J. E. Hunink, M. Parra, S. Contreras, P. Droogers, and E. Nicolás, "Effects of saline reclaimed waters and deficit irrigation on citrus physiology assessed by uav remote sensing," *Agricultural Water Management*, vol. 183, pp. 60–69, 2017.

[14] B. L. Machovina, K. J. Feeley, and B. J. Machovina, "Uav remote sensing of spatial variation in banana production," *Crop and Pasture Science*, vol. 67, no. 12, pp. 1281–1287, 2017.

[15] M. D. Larson, A. Simic Milas, R. K. Vincent, and J. E. Evans, "Multi-depth suspended sediment estimation using high-resolution remote-sensing uav in maumee river, ohio," *International Journal of Remote Sensing*, pp. 1–18, 2018.

[16] R. Bhola, N. H. Krishna, K. Ramesh, J. Senthilnath, and G. Anand, "Detection of the power lines in uav remote sensed images using spectral-spatial methods," *Journal of environmental management*, vol. 206, pp. 1233–1242, 2018.

[17] B. Coifman, M. McCord, R. G. Mishalani, and K. Redmill, "Surface transportation surveillance from unmanned aerial vehicles," in *Proc. of the 83rd Annual Meeting of the Transportation Research Board*, 2004.

[18] A. Puri, "A survey of unmanned aerial vehicles (uav) for traffic surveillance," *Department of computer science and engineering, University of South Florida*, pp. 1–29, 2005.

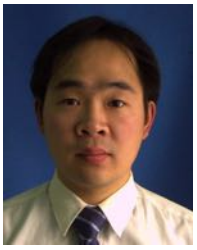
[19] T. Moranduzzo and F. Melgani, "Automatic car counting method for unmanned aerial vehicle images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 3, pp. 1635–1647, 2014.

[20] G. Salvo, L. Caruso, and A. Scordo, "Urban traffic analysis through an uav," *Procedia-Social and Behavioral Sciences*, vol. 111, pp. 1083–1091, 2014.

[21] K. Kanistras, G. Martins, M. J. Rutherford, and K. P. Valavanis, "Survey of unmanned aerial vehicles (uavs) for traffic monitoring," in *Handbook of unmanned aerial vehicles*. Springer, 2015, pp. 2643–2666.

[22] K. Gallagher and P. Lawrence, "Unmanned systems and managing from above: the practical implications of uavs for research applications

addressing urban sustainability," in *Urban Sustainability: Policy and Praxis*. Springer, 2016, pp. 217–232.
[23] M. A. Khan, W. Ectors, T. Bellemans, D. Janssens, and G. Wets, "Uav-based traffic analysis: A universal guiding framework based on literature survey," *Transportation Research Procedia*, vol. 22, pp. 541–550, 2017.
[24] M. Khan, W. Ectors, T. Bellemans, D. Janssens, and G. Wets, "Unmanned aerial vehicle-based traffic analysis: A case study for shockwave identification and flow parameters estimation at signalized intersections," *Remote Sensing*, vol. 10, no. 3, p. 458, 2018.
[25] B. Coifman, M. McCord, R. G. Mishalani, M. Iswalt, and Y. Ji, "Roadway traffic monitoring from an unmanned aerial vehicle," in *IEE Proceedings-Intelligent Transport Systems*, vol. 153, no. 1. IET, 2006, pp. 11–20.
[26] P. Skoglar, U. Orguner, D. Törnqvist, and F. Gustafsson, "Road target search and tracking with gimballed vision sensor on an unmanned aerial vehicle," *Remote sensing*, vol. 4, no. 7, pp. 2076–2111, 2012.
[27] T. Moranduzzo and F. Melgani, "Detecting cars in uav images with a catalog-based approach," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 52, no. 10, pp. 6356–6367, 2014.
[28] H.-Y. Cheng, C.-C. Weng, and Y.-Y. Chen, "Vehicle detection in aerial surveillance using dynamic bayesian networks," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 2152–2159, 2012.
[29] X. Chen, S. Xiang, C.-L. Liu, and C.-H. Pan, "Vehicle detection in satellite images by hybrid deep convolutional neural networks," *IEEE Geoscience and remote sensing letters*, vol. 11, no. 10, pp. 1797–1801, 2014.
[30] Z. Chen, C. Wang, C. Wen, X. Teng, Y. Chen, H. Guan, H. Luo, L. Cao, and J. Li, "Vehicle detection in high-resolution aerial images via sparse representation and superpixels," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 1, pp. 103–116, 2016.
[31] K. Liu and G. Mattyus, "Fast multiclass vehicle detection on aerial images," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 9, pp. 1938–1942, 2015.
[32] G. Mo and S. Zhang, "Vehicles detection in traffic flow," in *Natural Computation (ICNC), 2010 Sixth International Conference on*, vol. 2. IEEE, 2010, pp. 751–754.
[33] Z. Chen, T. Ellis, and S. A. Velastin, "Vehicle detection, tracking and classification in urban traffic," in *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*. IEEE, 2012, pp. 951–956.
[34] C. Zhang, H. Li, X. Wang, and X. Yang, "Cross-scene crowd counting via deep convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 833–841.
[35] Y. Zhang, D. Zhou, S. Chen, S. Gao, and Y. Ma, "Single-image crowd counting via multi-column convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 589–597.
[36] Z. Zhao, H. Li, R. Zhao, and X. Wang, "Crossing-line crowd counting with two-phase deep neural networks," in *European Conference on Computer Vision*. Springer, 2016, pp. 712–726.
[37] C. M. Bautista, C. A. Dy, M. I. Mañalac, R. A. Orbe, and M. Cordel, "Convolutional neural network for vehicle detection in low resolution traffic videos," in *Region 10 Symposium (TENSYMP), 2016 IEEE*. IEEE, 2016, pp. 277–281.
[38] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.
[39] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[41] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 3354–3361.
[42] L. Wen, D. Du, Z. Cai, Z. Lei, M.-C. Chang, H. Qi, J. Lim, M.-H. Yang, and S. Lyu, "Ua-detrac: A new benchmark and protocol for multi-object detection and tracking," *arXiv preprint arXiv:1511.04136*, 2015.
[43] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431–3440.
[44] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *European Conference on Computer Vision*. Springer, 2016, pp. 850–865.
[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
[46] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
[47] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
[48] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 1. IEEE, 2005, pp. 886–893.
[49] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.
[50] Itseez, "Open source computer vision library," https://github.com/itseez/opencv, 2015.
[51] M. Lin, Q. Chen, and S. Yan, "Network in network," *arXiv preprint arXiv:1312.4400*, 2013.
[52] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2147–2154.
[53] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
[54] A. Neubeck and L. Van Gool, "Efficient non-maximum suppression," in *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, vol. 3. IEEE, 2006, pp. 850–855.
[55] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3464–3468.
[56] J. H. Yoon, M.-H. Yang, J. Lim, and K.-J. Yoon, "Bayesian multi-object tracking using motion context from multiple objects," in *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*. IEEE, 2015, pp. 33–40.
[57] C. Dicle, O. I. Camps, and M. Sznaier, "The way they move: Tracking multiple targets with similar appearance," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 2304–2311.
[58] R. E. Kalman *et al.*, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.
[59] G. A. Mills-Tettey, A. Stentz, and M. B. Dias, "The dynamic hungarian algorithm for the assignment problem with changing costs," 2007.
[60] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.
[61] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.
[62] J. Leitloff, D. Rosenbaum, F. Kurz, O. Meynberg, and P. Reinartz, "An operational system for estimating road traffic information from aerial images," *Remote Sensing*, vol. 6, no. 11, pp. 11 315–11 341, 2014.
[63] X. Cao, C. Wu, J. Lan, P. Yan, and X. Li, "Vehicle detection and motion analysis in low-altitude airborne video under urban environment," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, no. 10, pp. 1522–1533, 2011.



**Jiasong Zhu** received his B.E. degree from Huazhong University of Science and Technology, M.E. degree from Wuhan University, and Ph.D. degree from The University of Hong Kong in 1997, 2003 and 2008, respectively. He is currently an Associate Professor with Shenzhen Key Laboratory of Spatial Information Smarting Sensing and Services, College of Civil Engineering, Shenzhen University, China.

His research interests include multi-sensor integration and data fusion, high resolution image processing, and GIS applications in urban planning and transportation.

**Ke Sun** is currently a Research Assistant in Shenzhen Key Laboratory of Spatial Information Smarting Sensing and Services, Shenzhen University and a PhD student in the University of Nottingham. He received the B.S. and M.S. degrees in Donghua University (Shanghai) and University of St Andrews (UK) in 2009 and 2013.

His research interests include Artificial Intelligence, Computer Vision and Natural Language Processing.

**Sen Jia** (M'13-SM'17) received his B.E. and Ph.D. degrees from College of Computer Science, Zhejiang University in 2002 and 2007, respectively. He is currently an Associate Professor with the College of Computer Science and Software Engineering, Shenzhen University, China.

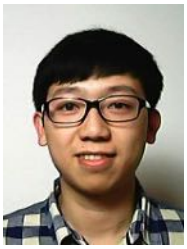
His research interests include hyperspectral image processing, signal and image processing, pattern recognition and machine learning.

**Qingquan Li** received the M.S. degree in engineering and the Ph.D. degree in photogrammetry and remote sensing from Wuhan University, Wuhan China, in 1988 and 1998, respectively. From 1988 to 1996, he was an Assistant Professor with Wuhan University, where he became an Associate professor from 1996 to 1998 and has been a Professor with the State Key Laboratory of Information Engineering in Surveying, Mapping, and Remote Sensing since 1998. He is currently the President of Shenzhen University and the Director of Shenzhen Key Laboratory of Spatial Information Smart Sensing and Services, Shenzhen University. He is an expert in Modern Traffic with the National 863 Plan and an Editorial Board Member of the Surveying and Mapping Journal and the Wuhan University Journal-Information Science Edition.

His research interests include photogrammetry, remote sensing, and intelligent transportation systems.

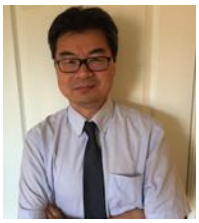
**Xianxu Hou** is currently a Post Doc in Guangdong Key Laboratory of Intelligent Information Processing, College of Information Engineering, Shenzhen University, China. He received the B.S. and M.S. degrees in China University of Mining and Technology (Beijing) in 2011 and 2014 and the PhD degree in the University of Nottingham in 2018.

His research interests include Deep Learning, Computer Vision and Natural Language Processing.



**Weidong Lin** is currently studying for M.S. degree with the College of Civil Engineering, Shenzhen University, China. He received the B.S. degree in Taiyuan University of Technology, China in 2016. His research interests include Intelligent Transportation System and Computer Vision.

**Bozhi Liu** Bozhi liu is currently a Research Fellow in Guangdong Key Laboratory of Intelligent Information Processing, College of Information Engineering, Shenzhen University, China. He received the B.S. degreee in Nanjing University of Science & Technology, and Master and PhD degrees in University of Nottingham, UK. His research interests include colour constancy, image processing and video enhancement.



**Guoping Qiu** received the B.S. degree from the University of Electronic Science and Technology of China, Chengdu, China, in 1984 and the Ph.D. degree from the University of Central Lancashire, Preston, U.K., in 1993. He is currently a Professor with Guangdong Key Laboratory of Intelligent Information Processing, College of Information Engineering, Shenzhen University, China and School of Computer Science, The University of Nottingham, UK.

His research interests include image processing, computer vision and deep learning.