Raw camera data object detectors: an optimisation for automotive processing and transmission

Pak Hung Chan*, Chuheng Wei*, Anthony Huggett, Valentina Donzella

***Abstract*— Whilst Deep Neural Networks have been developing swiftly, most of the research has been focused on RGB image. This type of image has been traditionally optimised for human vision. However, RGB data is a highly re-elaborated and interpolated version of the collected raw data (i.e. the sensor collects one value per pixel), but an RGB image for human viewing contains 3 values, for red, green, and blue. This processing through the ISP (Image Signal Processing) requires computational resource, time, power and obviously increases by a factor of three the amount of output data. This work investigates Deep Neural Network based detection using (for training and evaluation) Bayer data, generated in different ways, from a benchmarking automotive dataset (i.e. KITTI dataset). A Deep Neural Network (DNN) is deployed in unmodified form, and also modified to accept only single field images, such as Bayer frames. Eleven different re-trained version of the DNN are produced, and cross-evaluated across different data formats. The results demonstrate that the selected DNN has the same accuracy when evaluating RGB or Bayer data, without significant degradation in the perception (the variation of the Average Precision is <1%). Moreover, the colour filter array position and the colour correction matrix do not seem to contribute significantly to the DNN performance. This work demonstrates that Bayer data can be used for object detection in automotive without significant performance loss, and that the processing currently used in ISP can be avoided, allowing for more efficient sensing-perception systems.**

***Index Terms*—Bayer Data, Object Detection, Perception Sensors, Assisted and Automated Driving, Intelligent Vehicles.**

I. INTRODUCTION

WITH the advancement of computer hardware technology, deep learning-based artificial intelligence technologies are in rapid development and already being evaluated for or used in a wide range of applications, including assisted and automated driving (AAD) functions [1]. According to the J3016 standard, the Society of Automotive Engineers (SAE) defines six levels of driving automation (L0-L5) [2]. As functions on vehicles reach higher levels of automation (L3-L5), the ability to sense and make decisions based on the external environment becomes an increasingly essential capability. As a foundation for path planning, behavioural decisions, and motion control, environmental perception has become a key research topic in academia and industry [3]. The detection of traffic actors such as vehicles and pedestrians, and the implementation of real-time vehicle perception of the road conditions are important for the prevention of common types of traffic accident, as well as being important for the affirmation and expansion of automated driving applications in the near future [4-5].

Deep neural network (DNN) methods are well established techniques for detecting and classifying objects, and there is a rapidly growing body of work related to their use for road targets [6]. The R-CNN series and YOLO series DNNs are the most commonly used for object detection tasks, but there is a trade-off between detection accuracy and detection speed [7]. Until recently, most of the DNNs have been based (i.e. trained and tested) on images in a format of three colour channels, RGB (red, green, blue). In automotive, the RGB inputs to DNNs are the frames produced by HDR cameras, and in turn they are a processed version of the captured *raw sensor data*. The raw data corresponds to a value of light intensity collected per pixel through the colour filter array (CFA) used in the sensor, as shown in Fig. 1. Traditionally the CFA has been in the format of a R-G-G-B 2x2 repeated pixel matrix and optimised for human vision [8]. The conversion into RGB colour channels, through the colour pipeline and ISP (image signal processing) in the sensor, has been historically created to produce images looking pleasant and realistic to human viewers. However, this processing and manipulation might be not needed for machine learning and DNN-based perception, and this paper aims to explore if *raw* or *Bayer* data (i.e. one intensity per pixel or one intensity plus colour filter per pixel) can be used for perception without degrading DNN performance. Moreover, the use of raw data will reduce, roughly by a factor of three, the size of the total data that need to be transmitted into the processing algorithms (only one value per pixel instead of three), and will decrease the overall needed processing on chip [9].

The ISP pipeline includes several different manipulations of the raw data, including noise reduction, black level and white balance correction, colour balancing, gamma correction, dead pixels concealment, etc. Two general and similar diagrams for ISP processing are shown in Fig. 2. There are several different ways of implementing an ISP and they are outside the scope of this paper. Instead, this paper will focus on the contribution of two key steps in the colour pipeline, the *demosaicing* process and the *colour correction matrix*. The raw values captured by the camera are measured based on the light incoming from the real world in the specific position of the sensor pixel matrix

P.H. Chan, C. Wei, and V. Donzella are with WMG, University of Warwick, Coventry, United Kingdom.

Dr A Huggett is with onsemi, Greenwood House, Bracknell, RG12 2AA, UK (e-mail: anthony.huggett@onsemi.com).

* These authors equally contributed

Correspond to: Pak.Chan.1@warwick.ac.uk

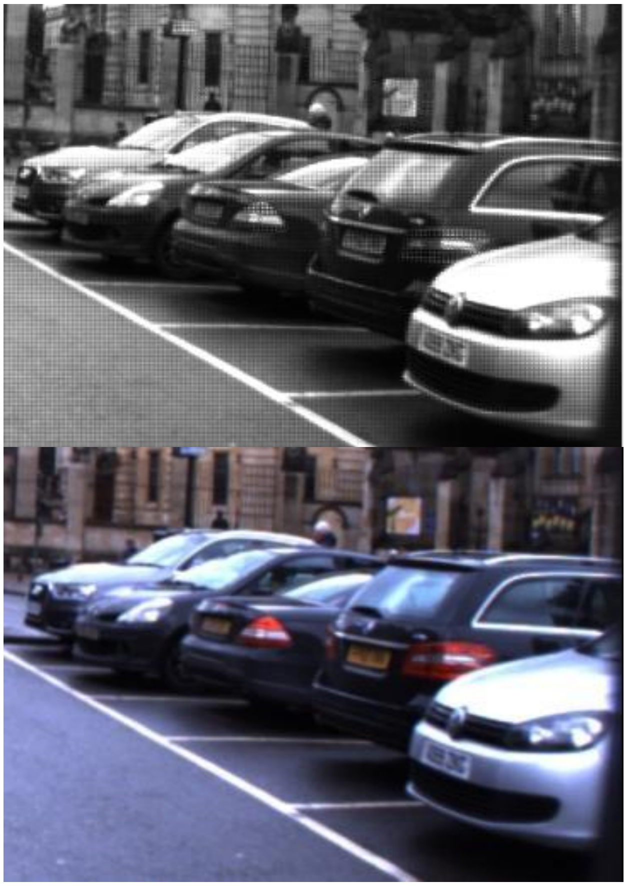


Fig. 1. A frame from the Oxford Robocar dataset [42]. Top is the raw image from the dataset, bottom is the same image after ISP processing and conversion in 3 colour channels.

(one intensity value per pixel). The demosaicing process then consumes these raw Bayer values to interpolate colour values for the three channels, each channel retaining the same resolution as the sensor pixel matrix. This process is implemented by interpolating the values of neighbouring pixels in the raw matrix, and different algorithms can be applied. The colour correction matrix is used to balance the gains for colour channels to ensure that the colour rendering of the images is realistic (for the human viewer). These 2 steps are selected as they are key to achieve 3 colour channels per pixel starting from the single intensity measured in that pixel.

A. Contributions

The main contributions of this paper are:

- Proposing and justifying different ways of ‘inverting’ RGB datasets into *‘Bayer’* datasets. This step can be key if big Bayer datasets need to be created from existing datasets for the re-training of state of the art (SOTA) DNN models;
- Evaluating the performance of a commonly used DNN based object detector using different *Bayer* datasets, either in training or testing or both;
- Comparing the DNN performance with Bayer data to the performance with ‘traditional’ RGB or grayscale images.
- Provide a simple method to create three channel *Bayer* image without distorting the information which can be used with exiting object detection neural networks.

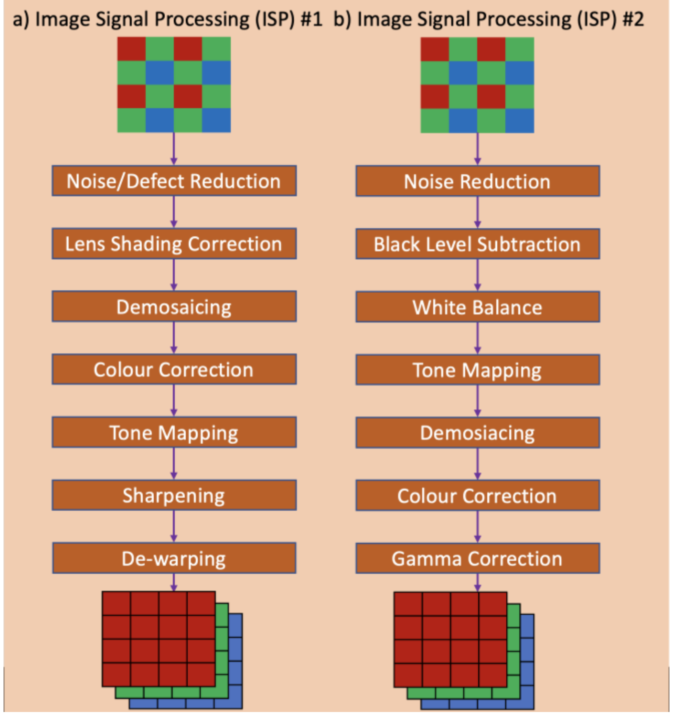


Fig. 2. a) Some fundamental processing steps in a generic colour pipeline in an imaging sensor, and b) the ISP blocks considered in [37]. In both cases, the output is a 3 colour channel image with the same resolution of the input Bayer data.

Overall the results show that it is possible to use *single colour channel ‘Bayer’ data* and to achieve detection performance comparable to the performance with traditional RBG data. This demonstration enables the processing needed on the sensor chip to be reduced, and moreover to transmit less camera data (minimally conditioned) to the vehicle processing unit(s), helping to address the previously presented data conundrum [9].

II. Related Work

A. Object Detection Methods

Some of the most important targets in traffic scenes are vehicles, pedestrians, and cyclists [10]. There are many studies regarding detection of these objects, and they can be broadly divided into three categories: traditional image processing detection methods, traditional machine learning methods, and deep neural network methods. Traditionally these studies have used RGB images, but recent work has also started to look into Bayer images, as reviewed in sec. II.B. The three categories are further discussed below.

1) Traditional image processing detection methods

These handcrafted object recognition methods, often based on regression, are difficult to apply to a wide range of real-world situations. Accurate results are difficult to obtain when weather conditions change, objects are obstructed or too dark, etc. Furthermore, different targets require different classifiers to be developed and real-time detection is impossible [11-15]

2) Traditional machine learning detection methods

Traditional machine learning detection algorithms for vehicles generally propose new vehicle-specific features or use other environmental information as an auxiliary detection method. Laopracha *et al.* [16] employed V-HOG features in combination with SVM kernel functions to detect vehicles, ensuring both accuracy and speed of the overall algorithm. Based on histograms of oriented gradients, Cao *et al.* developed a vehicle detection system based on the AdaBoost classifier, which can basically meet the requirements of real-time vehicle detection [17]. Similarly, pedestrian detection algorithms have been dominated by the introduction of new features, or multi-feature fusion methods. Bastian *et al.* presented the second-order aggregate channel features (SOACF) in pedestrian detection [18]. Based on a Random Forest ensemble, Marin et al. propose a method to combine multiple local experts in order to accurately detect pedestrians [19]. Takarli *et al.* proposed detecting pedestrians using a combination of global and local features [20].

However, the traditional detection methods based on artificially-designed features to train the classifier do not work well on vehicles in a variety of complex real-world conditions, such as low light, rainy days, motion blur, different positions of the vehicle on the image, and variations of the environment. For these reasons, they are not suitable for application in automated vehicles or advanced driving assistance system.

3) Deep neural network-based detection methods

There are three main categories for object detection neural networks, namely: one-stage, two-stage and transformers. One-stage networks such as YOLO and SSD have predefined overlapping regions of the image to detect and classify objects inside each region. A filtering process is performed to remove regions that are overlapping on one single object [21][22]. On the other hand, two-stage networks, such as RCNN and Fast-RCNN, contain a pipeline to perform both the region proposal and classification of the regions [23][24]. Comparatively, One-stage networks are generally faster, but will have lower accuracy compared to two-stage networks. Finally, vision transformers such as BERT or DETR, divide the images into patches and then search for relationship between pixels, however they require a lot of training data [25][26].

In order to ensure accurate detection, many automotive algorithms have been based on two-stage detection model, with Fast R-CNN and Faster R-CNN being the most used. Nguyen has proposed an improved Faster-RCNN vehicle detection algorithm to address the problems of large-scale variation and mutual occlusion in vehicle detection, with a 4% performance improvement compared to Faster-RCNN [27]. Rui et al. have developed the Feature Pyramid Network (PRN), based on the Faster R-CNN, for pedestrian detection and have proposed a method for combining multiple layers of features to detect small-sized pedestrians [28]. Zhang et al. have employed Faster R-CNN to implement pedestrian detection based on infrared images [29].

However, on-board object detection requires pressing real-time performance, and consequently, more and more one one-stage models have been investigated for automotive applications. Since the first version of YOLO proposed by Redmon *et al.*, there have been many refinements and improvements made, spanning several versions [21]. YOLOv3 is the most recent variant proposed by the original author [30]. YOLOv4 continues on from the base framework from YOLOv3, incorporating optimisation and improvement methods such as mosaic data augmentation, mish activation function and dropblock [31]. YOLOv4 has shown to have an improvement in performance of 10% in average precision and 12% in speed (frames per second) compared to YOLOv3 in MS COCO dataset (test-dev 2017) [31]. Jamiya and Rani have addressed the difficulty of balancing the speed and accuracy of current vehicle detection algorithms by enhancing YOLOv3 and incorporating the concept of Spatial Pyramid Pooling [32]. The proposed YOLO-SPP detection algorithm has shown good real-time performance, allowing timely responses in vehicle's warning systems. Moreover, Chao et al. have achieved an enhancement of detection of overlapped targets using the SSD algorithm and adding a rejection term to the DNN loss function [33]. There are further variants of YOLO from various groups, building on the spirit of the YOLO network, increasing performance in speed and accuracy by tweaking the architectures and incorporating new features [34][35].

B. Camera colour pipeline and DNNs

In addition to the above optimisation on the deep neural network algorithm, some researchers have started to investigate the possible impact of the quality/type of input data on the DNN performance.

Liu et al. examined the effect of camera parameters on the neural network and experimentally demonstrated that there was little difference between the detection of vehicles with monochrome and RGB images when using Mask R-CNN as the detection algorithm [36]. Some groups have investigated the effects of ISP-processed images on DNN detection, but the core of the work is not focused on automotive specific datasets and tasks. Hansen *et al.* have 'inverted' an ISP pipeline, considering a few building blocks of a generic ISP, as shown in Fig. 2b [37]. However, the Authors acknowledge that ISP is not invertible and therefore their inversion might add or modify the information contained in the original frames in unexpected ways. The ISP-processed images perform better than the 'inverted' raw images, according to the DNN used, however it is not clear how the 3 input network is adapted to use the 'raw' data. The Hansen *et al.* also present an ablation study based on the ISP blocks considered in their ISP model, stating that each block contributes to a performance enhancement for the DNN, except for the denoising. The Tone Mapping is reported as the block most beneficial to the accuracy of detection. The Authors also re-converted the 'inverted' raw into 'simulated' RGB images, and in this case DNN performance are still lower than on the original dataset [37]. This result highlights the need of investigating more the 'inversion' process before considering the achieved results reliable and generalisable. Lubana et al. propose a simplified version of ISP by selecting some arbitrary blocks in the colour pipeline and evaluating the detection of processed images on a trained deep neural network for an in-vehicle camera-based image recognition system [38]. The

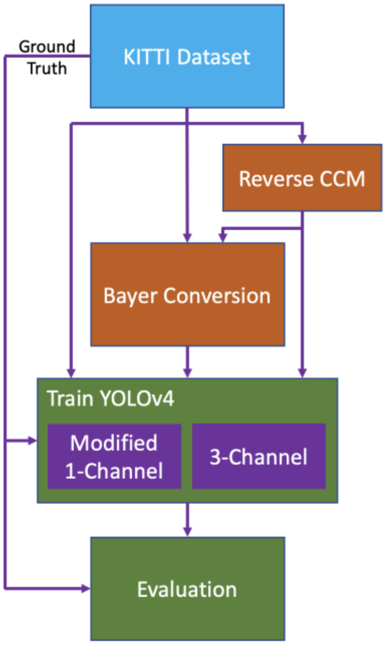


Fig. 3. Flow diagram showing the methodology followed for the presented experiments. Data for the experiments has been created starting from the KITTI dataset; 11 different variants of the dataset have been created and then used as training or testing data using YOLOv4 DNN.

results of their proposed algorithm show that image detection after their proposed algorithm is better than on raw images. However, the raw dataset used is very small (i.e. 225 images) and the DNN input architecture and training is not fully described. The full architecture is not described either in the recent paper by Chahill et al., but interestingly they use one-stage and two-stage object detector to evaluate different types of Bayer data, focusing on gamma-correction. However, the work only compares results of the DNN trained and tested on the same type of data, and the choice of gamma-correction is not fully convincing as a key step in the ISP pipeline [37]. Finally, recent work has investigated also end-to-end DNN methods to substitute the ISP pipeline, traditionally optimised for human vision, showing that for example these methods can outperform traditional ISP pipeline in the case of low light conditions and for machine learning applications [40-41]. These works demonstrate once more that image processing has been traditionally optimised for human vision, and better looking images do not necessarily produce the best results in the case of machine learning application.

This work builds on the recent work, aiming at understanding if commonly used DNN architectures, optimised for RGB images, can be easily re-used (e.g. just by transfer learning) with Bayer images, or a paradigm shift is needed in DNNs to use them.

III. Methodology

As previously stated, the processing introduced in cameras to convert a single value channel ‘image’ into three colour channels has been created for human vision. Single colour channel (*Bayer*) images roughly contain three times less data compared to three colour channel (RGB) images of the same size (in terms of resolution and bit depth), however the information content should be similar. This work investigates if *Bayer* information can be used for object detection without degrading the detection performance with respect to traditional RGB images. The following subsections explain the steps of our methodology, Fig. 3, and particularly our methods to convert an existing automotive dataset into *Bayer* images, allowing the ground truth bounding box information to be retained.

A. Dataset

Most commonly used automotive datasets generally provide three colour channel images. A recently released dataset by Oxford University, the RobotCar dataset, contains unrectified, 8-bit single colour channel *Bayer* image, top image in Fig. 1 [42]. However, this dataset does not provide object ground truth information needed if we want to use this data for neural network-based object detectors. Creating accurate labels for such a big dataset is a task outside of the aim of this work, hence an automotive benchmarking dataset, KITTI dataset, was chosen for the experiments and converted into *Bayer* image. The different methods to convert the dataset into *Bayer* are explained in Sec. III. B. In total, 8 three-channel datasets were generated, based on two different methods and four variants from colour correction matrix and colour filter array alignment, and 3 single-channel datasets.

The KITTI dataset was collected in Karlsruhe, Germany, and it contains sensor data collected on different road types [43]. It has been widely used as an automotive benchmarking dataset for different perception tasks such as object detection and segmentation. Amongst collected camera data, the dataset contains labelled camera data with eight classes: car, van, truck, pedestrian, person (sitting), cyclist, tram and misc. (indicating any other ‘objects’).

B. Conversion of Dataset

The selected KITTI dataset provides post-processed images which have been through the used camera image signal processing (ISP) and image rectification. ISP processing is not fully reversible and the specific ISP pipeline has not been released. In the work by Hansen et al., the Authors have tried to revert the ISP pipeline starting from RGB images, and then apply again the ISP process to create ‘simulated’ RGB images [37]. However these ‘simulated’ images had different performance with respect to the original RGB. In the hereby presented work, to avoid further modifications to the pre-processed data, we have created our *Bayer* datasets using as much as possible the values stored in the frames of the original KITTI dataset (from now on named ‘Original RGB’ dataset). We have also investigated and validated this approach by investigating the placement of the colour filter array (the CFA configuration is not known *a priori*). The different formats of the generated *Bayer* datasets are listed in Table I and described below.

In Table I, there are two *single channel formats* consisting of grayscale and Gray Bayer. The produced Gray Bayer dataset,

TABLE I

CONVERSION PROCESSES FOR CREATING THE DIFFERENT VERSIONS OF THE DATASET, WHERE SUBSCRIPTS INDICATE PIXEL POSITION IN AN ORIGINAL 2X2 BLOCK, BOLD LINES ARE USED TO CONTOUR EACH PIXEL (WITH ONE OR THREE INTENSITY VALUES), DOTTED LINE SPLIT COLOUR CHANNELS OF ONE PIXEL. CFA STANDS FOR COLOUR FILTER ARRAY, G_{AV} REPRESENTS THE AVERAGE OF THE $G_{1,2}$ AND $G_{2,1}$ GREEN PIXELS

Format Number	Format	Colour filter array	Comments
1	Original RGB	$R_{1,1}$ $G_{1,1}$ $B_{1,1}$ $R_{1,2}$ $G_{1,2}$ $B_{1,2}$ $R_{2,1}$ $G_{2,1}$ $B_{2,1}$ $R_{2,2}$ $G_{2,2}$ $B_{2,2}$	Original non modified KITTI frames, with **three colour channels** (3 colour values per each pixel)
2	Grayscale	Gray$_{1,1}$ Gray$_{1,2}$ Gray$_{2,1}$ Gray$_{2,2}$	This format is derived from the original dataset by applying a grayscale algorithm, resulting in a **single colour channel**
3	Gray Bayer	$R_{1,1}$ $G_{1,2}$ $G_{2,1}$ $B_{2,2}$	This format is composed by selecting only one colour channel (i.e. only one intensity) per pixel from 1) assuming an RGGB CFA, resulting in a **single colour channel**
4	Bayer 0-filled	$R_{1,1}$ 0 0 0 $G_{1,2}$ 0 0 $G_{2,1}$ 0 0 0 $B_{2,2}$	This format is composed by keeping an intensity value in a pixel for each channel only if it corresponds to the correct colour and position based on a RGGB CFA. Other pixels values are set to 0, resulting in **three colour channels**
4b	Bayer 0-filled (GRBG)	0 $G_{1,1}$ 0 $R_{1,2}$ 0 0 0 0 $B_{2,1}$ 0 $G_{2,2}$ 0	This format is a variant of 4, assuming GRBG CFA and resulting in **three colour channels**
4c	Bayer 0-filled (GBRG)	0 $G_{1,1}$ 0 0 0 $B_{1,2}$ $R_{2,1}$ 0 0 0 $G_{2,2}$ 0	This format is a minor variant of 4, with a GBRG CFA and resulting in **three colour channels**
4d	Bayer 0-filled (BGGR)	0 0 $B_{1,1}$ 0 $G_{1,2}$ 0 0 $G_{2,1}$ 0 $R_{2,2}$ 0 0	This format is a minor variant of 4, with a BGGR CFA and resulting in **three colour channels**
5	Bayer colour-filled	$R_{1,1}$ G_{av} $B_{2,2}$ $R_{1,1}$ $G_{1,2}$ $B_{2,2}$ $R_{1,1}$ $G_{2,1}$ $B_{2,2}$ $R_{1,1}$ G_{av} $B_{2,2}$	This format is composed by keeping an intensity value in a pixel for each channel only if it corresponds to the right colour and right position based on a RGGB CFA. For each 2x2 block, in the red and blue channels, the selected value is replicated in the other pixels. For the green channel, an average of the two green values is used to fill the 2 empty green values. **Three colour channels**

format 3, uses the colour channel value based on the colour filter for each pixel, similar to how in cameras the CFA creates one intensity value in each pixel. Format 2 dataset, grayscale, was created using a greyscaling algorithm which interpolates using the RGB values for every pixel and is used to act as a comparison against the generated Gray Bayer. In the case of single channel inputs we needed to modify YOLOv4, as explained in Sec. III.D. However, most modern object detection neural networks require three channel images as input. For the single channel Bayer and Greyscale formats, the selected neural network was simply altered in the input layer to accept single channel images, but the architecture was not optimised for this input.

To allow a comparison of *Bayer* performance against RGB images, three-colour channel *Bayer* images were created to use the neural network without modification. These three-colour channel *Bayer* image formats are designed to not modify the information content compared to single channel *Bayer* image. Format 4, Bayer 0-Filled, contains the same values as format 2, Gray Bayer, but split into the correct colour channels with the remaining pixels being filled with zero value. Although the information has not changed, the introduction of the zeros

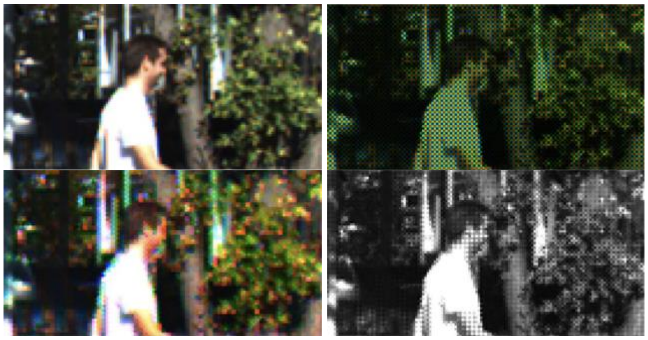


Fig. 4. Detail from a frame from the KITTI dataset and examples of applying different *Bayer* Conversions to it; original KITTI image top left, Bayer 0-Filled image top right, Bayer Colour-Filled bottom left and single channel gray Bayer bottom right.

might have an implication on the neural network detection (due to the zero patterns in the three colour channels). Hence, a second three channel *Bayer* image format dataset, Bayer Colour-Filled (format 5), was also generated. In this format, the pixel channels without values are instead filled with the corresponding pixel value of that channel in the 2x2 matrix, except green where it is filled with an average of the two values in the 2x2 matrix. Fig. 4 shows a detail from a frame of the KITTI dataset to visually compare 4 of the three channel variants generated and used in the presented experiments. As human consumers, the original frame (top left) is the most pleasant image, without 'abrupt' changes and with 'clear' details.

Moreover, the CFA placement is considered in this work. The alignment of the CFA for the KITTI dataset is not known. This work tests the different possible alignments of the CFA (i.e. RGGB, GRBG, GBRG, BGGR) to understand if it will have an effect on the evaluation of *Bayer* image with the selected neural networks. Three additional variants of the Bayer 0-Filled format were created based on the other alignment of the CFA, see formats 4b to 4d in Table I.

C. Colour Correction Matrix (CCM)

One critical part of the ISP is to perform colour correction through a colour correction matrix (CCM), generating a more natural image to the human visual system [37]. To investigate the effect of the CCM, a generic CCM, eq. 1, was inversed and applied to the dataset based on [44], see also Fig. 5. This process was performed to the original RGB dataset, Bayer 0-Filled and Gray Bayer which are the two formats most representative of a *Bayer* image generated. The inverse CCM was applied on the original image before the *Bayer* conversion was performed.

$$CCM = \begin{bmatrix} 1.6605 & -0.5876 & -0.0728 \\ -0.1246 & 1.1329 & -0.0083 \\ -0.0182 & -0.1006 & 1.1187 \end{bmatrix} \qquad (1)$$

D. YOLOv4

A one-stage network, namely a YOLOv4, is chosen for this work due to strict real-time requirements of the functions deployed for assisted and automated driving, see. Sec II. A. Several recent works have demonstrated that in the case of KITTI and automotive datasets, trends observed in one- and two-stage detectors are similar [39][45]. As a consequence, to reduce variability, our work focus on one architecture, but creating 11 trained DNN versions (see below) and cross-evaluate several datasets (i.e. original and different versions of one or three colour channel *Bayer*) per trained network. Our methodology can be followed and applied to any DNN architecture. To the best of our knowledge, this is the first study of this type. The input of the selected network requires RGB (3 channels) images, so as a part of this work a one channel input version of the YOLOv4 was created, but only the input layer of the network was modified, see Fig. 6. Part of the experiments are carried out with the DNN version with one channel input and part with three channels input, the generation of input data from the original RGB data is described in Table I. On the contrary, previous studies feeding 'Bayer' data to DNN do not specify how the 'Bayer' data nor the DNNs were modified, so they are not fully reproducible nor it is possible to deduce if the presented comparisons are fair. YOLOv4 was selected on the contrary on YOLOv5, because the latter does not have a publication to document the changes implemented, whereas newer YOLO versions are still not fully stable.

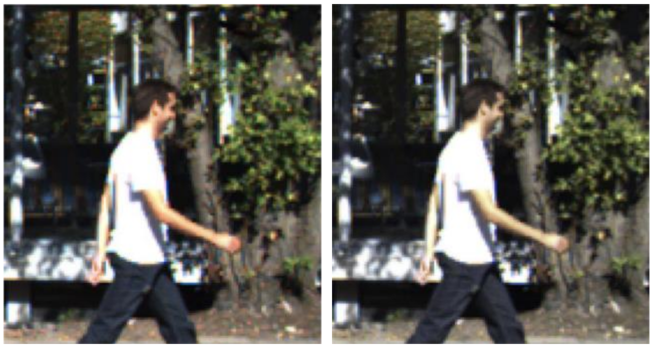


Fig. 5. Detail from a frame from the KITTI dataset: left is the original image, and right is the same image with the CCM inverse applied to it.

As a part of this work, the selected DNN was re-trained with different datasets, as described in Sec II. A-B, generating 11 versions of the re-trained network, of which 8 three channels input (i.e. original RGB, 0-filled Bayer, Colour-filled Bayer,

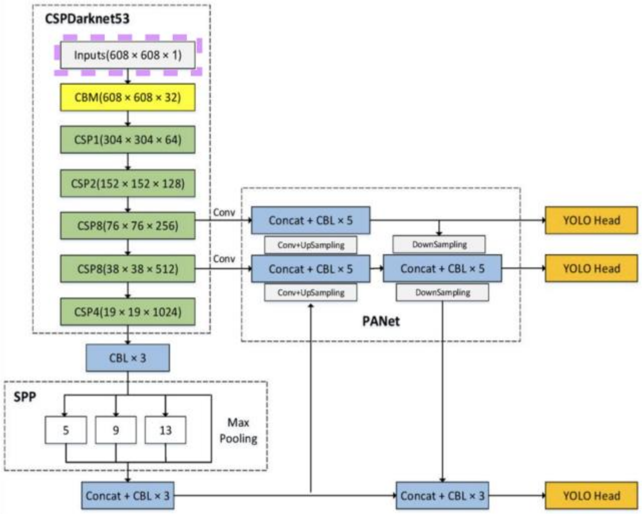


Fig. 6. The modified YOLOv4 architecture with the input layer (highlighted with dotted purple box) modified to accept one channel input for training and evaluating

original RGB no CCM, 0-filled Bayer no CCM, and 0-filled Bayer with 3 more variations of the CFA) and 3 one channel input (i.e. grayscale, gray Bayer, gray Bayer no CCM). These re-trained networks were then used to evaluate different three channel or one channel datasets, depending on the specific experiment. All combinations between training and testing are reported in Tables II-IV. The initial YOLO model parameters were initialised with small random numbers, close to 0. The model parameters were updated based on category loss, bounding box regression loss, and target confidence loss through 40 training epochs for every trained network.

E. Evaluation

The evaluation metrics selected in this paper are mAP$_{0.5}$ and mAP$_{[0.5:0.95]}$,. For mAP$_{0.5}$, the mean average precision across the classes is calculated, given an intersection over unit (IoU), between the predicated bounding box and ground truth, of 0.5. In mAP$_{[0.5:0.95]}$, the mAP is computed for each step between 0.5 and 0.95, with a step size of 0.05, and is then averaged. In the automotive field, mAP$_{0.5}$ show that objects are identified correctly, but can have a higher degree of uncertainty in the location of the object. On the other hand, mAP$_{[0.5:0.95]}$ evaluates with multiple steps of increasing IoUs, thus the score also considers how accurate is the location of the identified object in the image, which can be key for the safety of assisted and automated driving functions.

IV. Results

A total of 41 different pairings of the 11 re-trained DNNs and 11 generated datasets have been generated, of which 33 pairs related to the three input DNN and 8 pairs to the one input DNN version. Of the three single channel re-trained networks, the gray Bayer was cross-evaluated with respect to the grayscale (generating 4 sets of results) and with respect to the gray Bayer with no CCM (further 4 sets of results). For the 3 channel inputs, the Original, Bayer 0-filled and colour-filled were cross-evaluated (generating 9 sets of results); the 4 different configurations of CFA were cross-evaluated (generating 16 sets of results); and also Original and 0-filled versions were cross-evaluated with respect their inversion using the CCM (generating 8 more sets of results). All the results are reported in Tables II-IV.

A. Qualitative Results

Two adequately different frames were selected from the dataset to show the detection results, i.e. 000232.png and 000400.png, they are shown side by side in Fig. 7-8, where the detections when training and testing of the DNN are implemented with the same variants of the dataset. The selected frames are different in terms of visual content: Fig. 7. is not crowded but has 3 different types of road stakeholders, including one vulnerable stakeholder, i.e. the bike. Fig. 8. has several vehicles of different sizes and with different levels of occlusion. The detections and classifications with confidence scores for different objects are overlayed on the frames. Overall the detections look very similar in all the selected cases. In the case of Fig. 7-8, for each format of the dataset, the DNN has been re-trained and evaluated with the same data type, the detections on cross-evaluation are not reported, but they look similar to the reported results.

TABLE II
Table of results for the differently trained networks evaluated with the different datasets. Data formats are explained in Table I

Network Input Channels	Training set type	Evaluation set type	mAP$_{0.5}$	mAP $_{[0.5:0.95]}$
YOLOv4 with three channel input	Original RGB	Original RGB	**0.915**	**0.558**
		Bayer 0-Filled	0.828	0.507
		Bayer Colour-Filled	0.895	0.534
	Bayer 0-Filled	Original RGB	0.876	0.508
		Bayer 0-Filled	**0.897**	**0.522**
		Bayer Colour-Filled	0.876	0.506
	Bayer Colour-Filled	Original RGB	**0.912**	**0.552**
		Bayer 0-Filled	0.657	0.379
		Bayer Colour-Filled	**0.912**	0.549
YOLOv4 with one channel input	Grayscale	Grayscale	**0.907**	0.541
		Gray Bayer	0.903	**0.542**
	Gray Bayer	Grayscale	0.884	0.524
		Gray Bayer	**0.909**	**0.542**

B. Quantitative Results

The main results have been split into three tables (Tables II-IV) to identify three main aspects: comparing the detection performance of the DNNs trained with different types of data when evaluating RGB versus *Bayer* data; understanding the role of the position of the colour filter array on the results; analysing the role of the colour correction matrix. The top half of Table II shows the results in terms of the selected metrics, i.e. mAP$_{0.5}$ and mAP$_{[0.5:0.95]}$, when the YOLOv4 network accepts a three colour channel input, and the bottom half presents the results of the modified YOLOv4 accepting only a single colour channel input. For three channel input version, YOLO has been re-trained with original RGB data, Bayer 0-filled, and Bayer colour-filled and cross-evaluated across these formats, for the one input the network has been re-trained with Grayscale and Gray Bayer and cross-evaluated. The highest metrics values for each trained network have been highlighted

TABLE III
EVALUATION RESULTS COMPARING DIFFERENT COLOUR FILTER ARRAY PLACEMENT (TABLE I) WHEN GENERATING THE BAYER 0-FILLED

Training Set	Evaluation Set Bayer 0-filled	$mAP_{[0.5]}$	$mAP_{[0.5:0.95]}$
Bayer 0-filled (4)	RGGB (4)	**0.897**	**0.522**
	GRBG (4b)	0.887	0.510
	GBRG (4c)	0.891	0.517
	BGGR (4d)	**0.897**	0.513
Bayer 0-filled GRBG (4b)	RGGB (4)	0.894	0.521
	GRBG (4b)	0.891	0.520
	GBRG (4c)	0.889	**0.529**
	BGGR (4d)	**0.897**	0.524
Bayer 0-filled GBRG (4c)	RGGB (4)	0.895	0.535
	GRBG (4b)	0.895	0.528
	GBRG (4c)	0.890	0.517
	BGGR (4d)	**0.899**	**0.537**
Bayer 0-filled BGGR (4d)	RGGB (4)	0.887	0.527
	GRBG (4b)	0.886	**0.531**
	GBRG (4c)	0.884	0.518
	BGGR (4d)	**0.893**	0.524

in bold, and the best metrics across the different combinations show comparable performance within 5%.

Table III presents the results of the DNN re-trained with the Bayer 0-filled considering 4 different RGB CFA placement, as shown in Table I (i.e. 4, 4b, 4c, 4d). The 4 generated three channel networks have been cross-evaluated with testing data generated with the 4 CFA placements too. The highest metrics values for each trained network have been highlighted in bold, and the best metrics across the different combinations show comparable performance within 2%, with $mAP_{0.5}$ varying of less than 1%.

Finally, Table IV compares the results when trying to remove the effect of the CCM processing from the original RGB data. In this case 4 different three input channel networks have been re-trained with the original RGB data, original RGB before CCM (cross-evaluating these two formats), Bayer 0-filled and Bayer 0-filled without CCM (again cross-evaluating these two formats), and then 2 different one input channel networks have been trained with Gray Bayer and Gray Bayer before CCM (cross-evaluating too). The highest metrics values for each trained network have been highlighted in bold, and the best metrics across the different combinations show comparable performance within 2%, with $mAP_{0.5}$ varying of less than 1%.

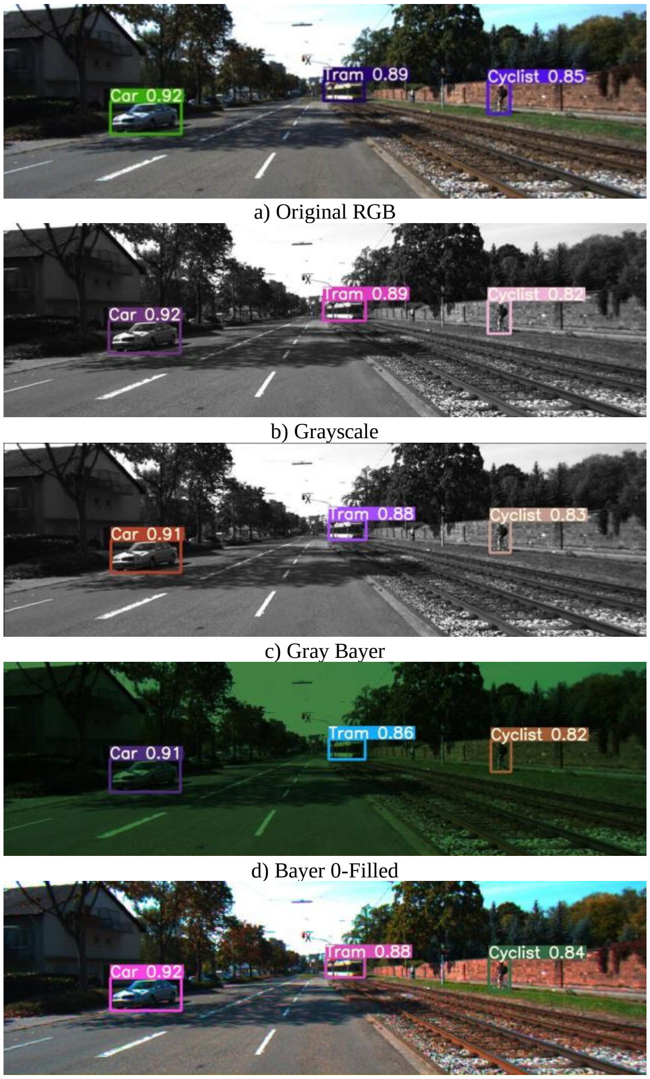
V. DISCUSSION

In terms of the qualitative results (Figs. 7-8), the detections for the selected pairs training-testing are extremely similar in the two frames, with small variations of the confidence scores. An interesting aspect is that even in the case of occluded vehicles and vulnerable road users (e.g. the cyclist), the DNN is able to classify them correctly for all the re-training and all the data formats used. Overall the detection of vulnerable road users and small objects in the frame does not seem detrimentally affected when using the different formats of Bayer data, and performance are very close to the performance of the original RGB data.

TABLE IV
EVALUATION RESULTS COMPARING THE REMOVAL OF A PSEUDO CCM

Training Set Type	Evaluation Set Type	$mAP_{[0.5]}$	$mAP_{[0.5:0.95]}$
Original RGB	Original RGB	**0.915**	**0.558**
	Original RGB No CCM	0.912	0.555
Original RGB No CCM	Original RGB	0.913	0.554
	Original RGB No CCM	**0.916**	**0.562**
Bayer 0-Filled	Bayer 0-Filled	**0.897**	**0.522**
	Bayer 0-Filled No CCM	0.877	0.511
Bayer 0-Filled No CCM	Bayer 0-Filled	**0.902**	**0.540**
	Bayer 0-Filled No CCM	0.894	0.533
Gray Bayer	Gray Bayer	**0.909**	**0.542**
	Gray Bayer No CCM	0.874	0.512
Gray Bayer No CCM	Gray Bayer	0.893	**0.533**
	Gray Bayer No CCM	**0.897**	0.528

These results are further confirmed by the values reported in the Tables. II-IV. In all the training-testing combinations, the best performance with each network trained with a different variant of the dataset ranged between 0.893 to 0.916 for $mAP_{0.5}$, and 0.522 to 0.568 for $mAP_{[0.5:0.95]}$. These results suggest that when using DNN based object detectors, there are minor performance variations in the 'detection' using different representation of the data. However, the accuracy of the bounding boxes (i.e. position and size) may suffer sightly more. On a high level, the different ways of representing Bayer information in Table II contained the same information, derived from the original dataset, but are arranged differently. Hence, the achieved values demonstrate that the DNN can cope with small changes in how the information is fed to the Network. However, it seems that the Bayer 0-filled version yields to the worst performance (i.e. 2% lower than the RGB-RGB training/evaluation version), this performance decrease might de due to zeros patterns in the input data hindering the network convolution and feature extraction. The Bayer colour-filled trained Network has a very interesting property: the detection performance in terms of $mAP_{0.5}$ is the same when evaluating the Bayer colour-filled and the original RGB data and only 0.3% different from the RGB-RGB DNN performance. It means that actually the performance when using this format of data are indistinguishable from traditional RGB based DNNs,



a) Original RGB

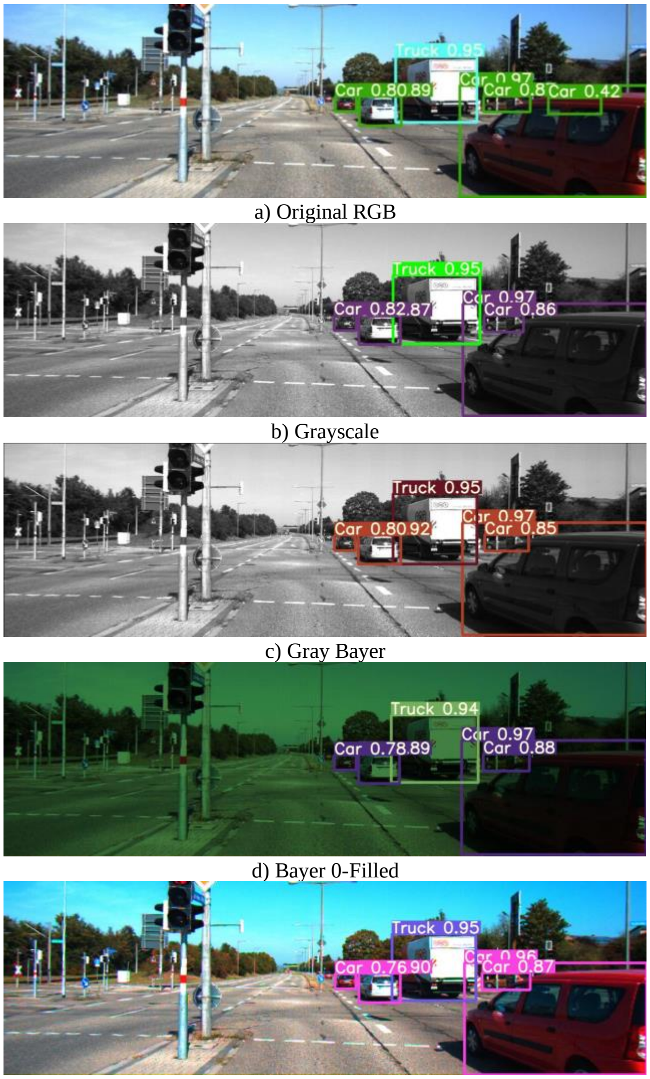
b) Grayscale

c) Gray Bayer

d) Bayer 0-Filled

e) Bayer Colour-Filled

Fig. 7. Objects detected by the trained network overlayed onto the frame 000232.png. The training and evaluating dataset uses the same image formats



a) Original RGB

b) Grayscale

c) Gray Bayer

d) Bayer 0-Filled

e) Bayer Colour-Filled

Fig. 8. Objects detected by the trained network overlayed onto the frame 000400.png. The training and evaluating dataset uses the same image formats

but also that hyperparameter tuning can be implemented for the colour-filled Bayer re-trained YOLO, yielding to even higher performance, further enhancing mAP$_{[0.5:0.95]}$. These results imply that state-of-the-art networks can be re-used and further optimised for consumption of Bayer data. This aspect will enable an immediate reduction of bandwidth required for camera data transmission on traditional vehicle communication networks, in the sense that camera data can be transmitted as non-processed single channel Bayer, and then 'colour-filled' to three channels in the DNN input stage. Moreover, recent work has mentioned that the use of raw images can reduce overall sensor power consumption (up to 35%) and the processing time of one sixth of the framerate, so the use of Bayer images in automotive can bring a significant optimisation when using cameras for assisted and automated driving functions [39].

For the network adapted for single channel input, using greyscale image and gray Bayer images performed extre mely similarly. However, the grey Bayer (training and evaluation) DNN performed the best for both mAP$_{[0.5]}$ and mAP$_{[0.5:0.95]}$. Additionally, YOLO trained and evaluated with grey Bayer performed very similarly to the network trained and evaluated with the original RGB, with a negligible decrease of 0.6% in mAP$_{0.5}$ and of 1.6% in mAP$_{[0.5:0.95]}$. These results show that single channel Bayer images can be a promising direction for research, with possible performance gain through slimmer networks (smaller inputs), optimised network architectures, and hyperparameter tuning.

In addition, as mentioned earlier, the exact colour filter array placement in the image is not known. This issue could affect the conversion of the dataset, see 4 to 4d formats in Table I. To consider how the exact CFA placement affect the DNN output, we have trained and cross-evaluated YOLO with the four CFA datasets. The results are recorded in Table III. Due to the demosaicing process used to produce the RGB image, every pixel channel has a dependence and relation to neighbouring pixels. Hence, although a different CFA pattern is applied, there are no major differences in pixel value patterns and image features should still be recognisable. Therefore, the colour filter array orientation does not play a large factor this work.

Finally, Table IV compares some of the best performing data formats with their version pre-CCM. For the original RBG data, CCM do not seem to play a significant role in the performance.

This is similar for the Bayer 0-filled and the Gray Bayer. This outcome shows once more that the processing in the ISP is not really optimised for DNN perception, and therefore it can be removed whereas more effort is allocated into converting existing DNNs to use Bayer data and maximising their performance.

VI. CONCLUSION

This paper presents a study on the use and re-training of DNN based object detectors to consume Bayer data instead of traditional RGB data, without any modifications to the neural network architecture. Moreover, with minimal adjustments, a DNN has been also converted to use single channel images, and when using the Gray Bayer dataset, the DNN performance have been almost identical (with a variation of 0.6%) to the traditional version of the Neural Network. The placement of CFA on sensors and the role of CCM have been analysed and discussed, and overall their effect seems marginal for the Network performance. The achieved results show that whilst the internal processing on camera sensors has been optimised for human vision, most of the implemented processing is not really needed for DNN based perception, and specifically for object detection. These findings not only pave the way for the re-use of current DNN in order to consume Bayer data, but also open the possibility to develop optimised architectures to use Bayer. In turns these achievements can improve the safety of future assisted and automated driving functions and also their efficiency, in terms of less sensor data to be transmitted to the vehicle processing unit(s), less sensor power consumption, and reduction of latency due to ISP.

ACKNOWLEDGMENT

Dr Donzella acknowledges that this work was supported by the Royal Academy of Engineering under the Industrial Fellowships scheme. This work is also partially funded by the European Union (grant no. 101069576). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Climate, Infrastructure and Environment Executive Agency (CINEA). Neither the European Union nor the granting authority can be held responsible for them. The UK participant (WMG) in this project is supported by Innovate UK (contract no. 10045139). The Authors wish to acknowledge the High Value Manufacturing CATAPULT.

REFERENCE

[2] M. Cococcioni, F. Rossi, E. Ruffaldi, S. Saponara, and B. D. de Dinechin, "Novel arithmetics in deep neural networks signal processing for autonomous driving: Challenges and opportunities," *IEEE Signal Processing Magazine,* vol. 38, no. 1, pp. 97-110, 2020.

[1] SAE, "Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles," J3016_202104, 2021.

[2] H. Zhu, K. V. Yuen, L. Mihaylova, and H. Leung, "Overview of Environment Perception for Intelligent Vehicles," *IEEE Transactions on Intelligent Transportation Systems,* vol. 18, no. 10, pp. 2584-2601, 2017, doi: 10.1109/TITS.2017.2658662.

[3] H. Guo, Y. Zhang, S. Cai, and X. Chen, "Effects of Level 3 Automated Vehicle Drivers' Fatigue on Their Take-Over Behaviour: A Literature Review," *Journal of Advanced Transportation,* vol. 2021, 2021.

[4] T. Cohen and C. Cavoli, "Automated vehicles: Exploring possible consequences of government (non) intervention for congestion and accessibility," *Transport reviews,* vol. 39, no. 1, pp. 129-151, 2019.

[5] M. Abbasi, A. Shahraki, and A. Taherkordi, "Deep learning for network traffic monitoring and analysis (NTMA): A survey," *Computer Communications,* vol. 170, pp. 19-41, 2021.

[6] Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, "Object Detection in 20 Years: A Survey," in *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257-276, March 2023, doi: 10.1109/JPROC.2023.3238524.

[7] C. -P. Hsu *et al*., "A Review and Perspective on Optical Phased Array for Automotive LiDAR," in *IEEE Journal of Selected Topics in Quantum Electronics*, vol. 27, no. 1, pp. 1-16, Jan.-Feb. 2021, Art no. 8300416, doi: 10.1109/JSTQE.2020.3022948.

[8] P. H. Chan, A. Huggett, G. Souvalioti, P. Jennings, and V. Donzella, "Influence of AVC and HEVC compression on detection of vehicles through Faster R-CNN," 2022, submitted to *IEEE Transactions on Intelligent Transportation Systems* (under review).

[9] Z. Liu, T. Lian, J. Farrell, and B. A. Wandell, "Neural network generalization: The impact of camera parameters," *IEEE Access,* vol. 8, pp. 10443-10454, 2020.

[10] A. Bensrhair, M. Bertozzi, A. Broggi, P. Miche, S. Mousset, and G. Toulminet, "A cooperative approach to vision-based vehicle detection," in *ITSC 2001. 2001 IEEE Intelligent Transportation Systems. Proceedings (Cat. No. 01TH8585)*, 2001: IEEE, pp. 207-212.

[11] J. M. Collado, C. Hilario, A. De la Escalera, and J. M. Armingol, "Model based vehicle detection for intelligent vehicles," in *IEEE Intelligent Vehicles Symposium, 2004*, 2004: IEEE, pp. 572-577.

[12] Y. Chong, W. Chen, Z. Li, W. H. Lam, C. Zheng, and Q. Li, "Integrated real-time vision-based preceding vehicle detection in urban roads," *Neurocomputing,* vol. 116, pp. 144-149, 2013.

[13] R. A. Hadi, G. Sulong, and L. E. George, "Vehicle detection and tracking techniques: a concise review," in *Signal & Image Processing: An International Journal, vol. 5, no. 3,* 2014.

[14] A. Haselhoff and A. Kummert, "A vehicle detection system based on haar and triangle features," in *2009 IEEE intelligent vehicles symposium*, 2009: IEEE, pp. 261-266.

[15] N. Laopracha, T. Thongkrau, K. Sunat, P. Songrum, and R. Chamchong, "Improving vehicle detection by adapting parameters of HOG and kernel functions of SVM," in *2014 International Computer Science and Engineering Conference (ICSEC)*, 2014: IEEE, pp. 372-377.

[16] X. Cao, C. Wu, P. Yan, and X. Li, "Linear SVM classification using boosting HOG features for vehicle detection in low-altitude airborne videos," in *2011 18th IEEE International Conference on Image Processing*, 11-14 Sept. 2011, 2011, pp. 2421-2424, doi: 10.1109/ICIP.2011.6116132.

[17] B. T. Bastian and J. CV, "Pedestrian detection using first-and second-order aggregate channel features," *International Journal of Multimedia Information Retrieval,* vol. 8, no. 2, pp. 127-133, 2019.

[18] J. Marin, D. Vázquez, A. M. López, J. Amores, and B. Leibe, "Random forests of local experts for pedestrian detection," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2592-2599.

[19] F. Takarli, A. Aghagolzadeh, and H. Seyedarabi, "Combination of high-level features with low-level features for detection of pedestrian," *Signal, Image and Video Processing,* vol. 10, no. 1, pp. 93-101, 2016.

[20] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[21] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. -Y. Fu and A. C. Berg, "SSD: Single Short MultiBox Detector," in *Proceedings of the European Conference on Computer Vision (ECCV) (2016)*, vol 9905, pp21-37.

[22] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," *2014 IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, USA, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.

[23] R. Girshick, "Fast R-CNN," *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.

[24] J. Devlin, M. -W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Comptutational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp.4171-4186, 2019.

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov and S. Zagoruyko, “End-to-End Object detection with Transformers,” in *Proceedings of the European Conference on Computer Vision (ECCV) (2020)*, col. 12346, pp. 213-229.

[26] H. Nguyen, "Improving Faster R-CNN Framework for Fast Vehicle Detection," *Mathematical Problems in Engineering,* vol. 2019, p. 3808064, 2019/11/22 2019, doi: 10.1155/2019/3808064.

[27] T. Rui, J. Fei, Y. Zhou, H. Fang, and J. Zhu, "Pedestrian detection based on deep convolutional neural network," *Computer Engineering and applications,* vol. 52, no. 13, pp. 162-166, 2016.

[28] L. Zhang, L. Lin, X. Liang, and K. He, "Is Faster R-CNN Doing Well for Pedestrian Detection?," in *Proceedings of the European Conference on Computer Vision (ECCV) (2016)*, vol 9906, pp 443-457.

[29] J. Redmond and A. Farhadi, “YOLOv3: An Incremental Improvement,” *arXiv preprint arXIV: 1804.02767*, 2018.

[30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[31] S. S. Jamiya and P. E. Rani, ‘‘LittleYOLO-SPP: A delicate real-time vehicle detection algorithm,’’ Optik, vol. 225, Jan. 2021, Art. no. 165818, doi: 10.1016/j.ijleo.2020.165818.

[32] J. Cao *et al.*, "Front vehicle detection algorithm for smart car based on improved SSD model," *Sensors,* vol. 20, no. 16, p. 4646, 2020.

[33] C. Li *et al.*, "YOLOv6: A Single_stage Object Detection Framework for Industrial Applications," *arXiv preprint arXiv:2209.02976*, 2022.

[34] C. -Y. Wang, A. Bochkovskiy and H. -Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” *arXiv preprint arXiv:2207.026696,* 2022.

[35] Z. Liu, T. Lian, J. Farrell, and B. A. Wandell, "Neural network generalization: The impact of camera parameters," *IEEE Access,* vol. 8, pp. 10443-10454, 2020.

[36] P. Hansen *et al.*, "ISP4ML: The role of image signal processing in efficient deep learning vision systems," in *2020 25th International Conference on Pattern Recognition (ICPR)*, 2021: IEEE, pp. 2438-2445.

[37] E. S. Lubana, R. P. Dick, V. Aggarwal, and P. M. Pradhan, "Minimalistic image signal processing for deep learning applications," in *2019 IEEE International Conference on Image Processing (ICIP)*, 2019: IEEE, pp. 4165-4169.

[38] J. Cahill *et al*., "Exploring the Viability of Bypassing the Image Signal Processor for CNN-Based Object Detection in Autonomous Vehicles," in *IEEE Access*, vol. 11, pp. 42302-42313, 2023, doi: 10.1109/ACCESS.2023.3270710.

[39] S. Diamond, V. Sitzmann, F. Julca-Aguilar, S. Boyd, G. Wetzstein amd F. Heide, “Dirty pixels: Towards end-to-end image processing and perception,” *ACM Transactions on Graphics (TOG)*, vol. 40, no, 3, pp. 1-15, 2021.

[40] E. Tseng, A. Mosleh, F. Mannan, K. St-Arnaud, A. Sharma, Y. Peng, A. Braun, D. Nowrouzezahrai, J. -F. Lalonde, F. Heide, “Differentiable compound optics and processing pipeline optimization for end-to-end camera design,” *ACM Transactions on Graphics (TOG)*, vol. 40, no. 2, pp. 1-19, 2021.

[41] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *The International Journal of Robotics Research,* vol. 36, no. 1, pp. 3-15, 2017.

[42] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research,* vol. 32, no. 11, pp. 1231-1237, 2013.

[43] Z. Roman. "Color correction with matrix transformation." https://support.medialooks.com/hc/en-us/articles/360030737152-Color-correction-with-matrix-transformation (accessed 5.10, 2022).

[44] B. Li, P. H. Chan, G. Baris, M. D. Higgins and V. Donzella, "Analysis of Automotive Camera Sensor Noise Factors and Impact on Object Detection," in *IEEE Sensors Journal*, vol. 22, no. 22, pp. 22210-22219, 15 Nov.15, 2022, doi: 10.1109/JSEN.2022.3211406.